PRIJS ƒ 7.25/Bfr. 135

Commodore INFO

EXTRA: Het uitgebreide Amiga-boeken overzicht

ONAFHANKELIJK BLAD VOOR COMMODORE GEBRUIKERS

JAARGANG 5, NO. 2, MAART 1988



LISTINGS

- Memory
- Windowla
- Renumber
- Ballonexpeditie
- Minisheet
- Lucifers C 16
- Klinkers C 16
- Regenbui C 16

Functie-analyse (2)

Smooth Scrolling

Tekening-conversie

Amiga 3D

Leerprogramma's

Vaste rubrieken

Basic Miniatuurtjes

Machinetaal

Basic cursus

Commodore Info

Verschijnt 8x per jaar
Jaargang 5, no. 2, maart 1988

Uitgave:
Sala Communications/SAC

Uitgever:
Drs. J. Taverne

Redactie:
Ir. L. Sala hoofdredacteur
J. Bodzinga adj. hoofdredacteur

J. Boers, drs. M. de Rooij, drs. H. Zoete, H. Smeenk, drs. U. Schuurmans, K. v.d. Vlies, R. Goudriaan, B. Munniksma, P.C. Broekhuizen.

Redactiesecretariaat:
R. van Zalingen

Strip: Bert Tier
Illustraties: Ben van Mierlo
Ymmot

Advertentie-exploitatie:
Ing. V. Sala
Weesperstraat 103
1018 VN Amsterdam
tel. 020-273198

Redactie adres:
Postbus 112
1260 AC Blaricum
tel. 02152-65695

Listingtelefoon:
(ma: 17.00-21.00) 02155-25162

Abonnementen en administratie:
Postbus 43048
1009 ZA Amsterdam
tel. 020-248006

Vragen betreffende abonnementen ontvangen wij bij voorkeur schriftelijk, met meesturen van het omslagetiket.

Abonnement:
Voor 8 nummers ƒ 47,50 of BFR. 950 per jaar. Betaling op giro 1585491 t.n.v. SAC/Commodore-Info.
Oude nummers kunt U alleen krijgen bij vooruitbetaling van ƒ 6,75 op de bovenstaande rekening.
Ook telefonische opgave voor een abonnement is mogelijk. Bel GRATIS 06-02242222 (teleservice), elke dag tot 20.20 uur (dus ook in het weekend). België: 115555, dagelijks tot 22.00 uur. Deze telefoonnummers zijn alleen bedoeld voor opgave van NIEUWE abonnementen.

Druk: Verweij, Mijdrecht
NDB, Zoeterwoude

Distributie:
In Nederland: Betapress, Gilze
In België: AMP, Brussel

ISSN: 0169-3085

Inhoud van dit nummer

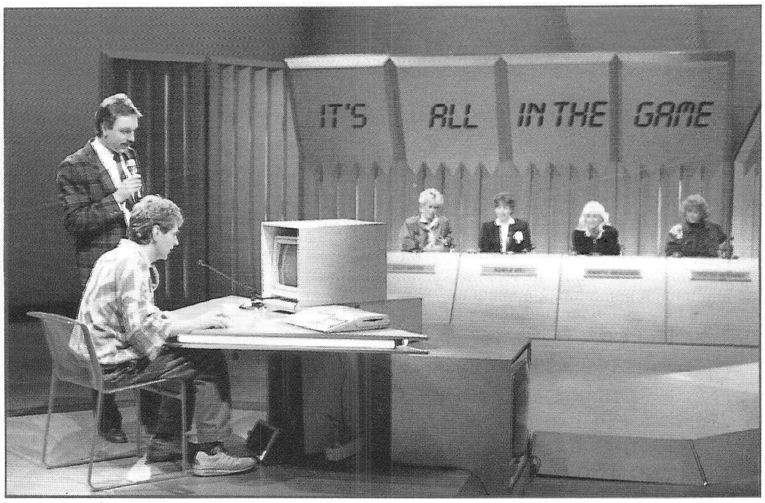
Redactioneel

Van Commodore krijgen we de laatste tijd weer heel goede berichten. De omzet is weer gestegen en er wordt ook weer flink winst gemaakt. De goede verkopen van de Amiga blijken daar een belangrijke oorzaak van te zijn. De 500, 1000 en 2000 zijn echte succesnummer met langzamerhand een trouwe schare volgelingen. De reakties op de lijsten met Amiga software in ons vorige nummer waren dan ook heel positief. Helaas is niet alles in ons land ook makkelijk te verkrijgen. Zelfs de public domain software, waarvoor het enorme Amiga aanbod aan die machine een eigen accent begint te geven, is niet erg gemakkelijk te krijgen. Daarom straten we met een nieuw lezersservice initiatief, Amiga Busware. Dat is Public Domain Software, die we in samenwerking met o.a. Courbois Software selekteren en uitbrengen. De prijs, inclusief 20% BTW en verzendkosten, is ƒ 11,- per schijfje. Voor de 64 en 128 gebruikers is er een belangrijk nieuwtje, Commodore heeft een nieuwe diskdrive uitgebracht, ditmaal ook met 3,5 inch schijfjes. En wat een vooruitgang, zoals we constateerden toen we met veel moeite zo'n 1581 te pakken kregen. Razendsnel, met 800 KB bepaald volwassen van capaciteit en veel handiger dan de oudere drive-types. Er zijn ook veel nieuwe functies, (zo kan er o.a. met sub-directories gewerkt worden). De lokale disk-intelligentie zorgt ervoor, dat op deze manier de 64 en 128 weer helemaal bij de tijd zijn. We verwachten op de komende Hannover Messe/Cebit overigens nog wel meer CBM nieuws.

Luc Sala

Computerspelletjes winnen nog steeds aan populariteit. Dat blijkt ook uit het spelprogramma 'It's all in the Game' van de NCRV. Een uniek samenwerkingsprojekt gaat er nu voor zorgen, dat de TV-games uit dat programma ook thuis kunnen worden gespeeld.

It's all in the Game



Presentator Henk Mouwe met enkele kandidaten.

De NCRV, René Stokvis Produkties en Sala Communications hebben gemeenschappelijk een uniek projekt opgezet. De computergames, die dit spelprogramma de revue zullen passeren, kunnen op diskette besteld worden, waardoor de kijkers nu ook zelf aktief met de spelletjes 'in de slag' kunnen.

It's All in the Game is vanaf dit seizoen elke maand op maandagavond te zien na het journaal van acht uur. Presentator Henk Mouwe test dan de behendigheid van een achttal kandidaten (in twee teams) op het gebied van computerspelletjes. De formule blijkt uitstekend aan te slaan getuige de kijkcijfers.

Thuis spelen

Om meer mensen de mogeljkheid te bieden de spelletjes te spelen, is nu besloten de games ook op een diskette uit te brengen. In overleg met René Stokvis Produkties (de bedenkers van dit programma) en de NCRV zal Sala Communications (uitgever van diverse computer-magazines, waaronder Commodore Info) de produktie en verkoop van diskettes gaan verzorgen.
Zo kunnen ook de thuisblijvers en de fervente computergame-spelers hun eigen kwaliteiten testen met dezelfde spelletjes als in de programma's aan de orde zullen komen.

Diskette I (Alleen voor C-64/128)

Op de eerste diskette staan de volgende spelletjes:

° **Balletje de Luxe**
één balletje, zes dopjes, en maar raden...

° **Schuif Schuif**
3 logo's, 8 dopjes. Schuiven in een oogwenk.

° **Medeklinkers**
16 cotegorieën, vele honderden woorden, maar... géén medeklinkers!

Aanvragen

Bestellen van de diskettes met daarop de games kan eenvoudig door *f*19,95 over te maken op gironummer **1585491** t.v.n. Sala Communications te Amsterdam met vermelding: 'It's all in the game'. U kunt ook een girobetaalkaart opsturen naar Postbus 5570, 1007 AN Amsterdam (ook hierop niet vergeten te zeggen, welke diskette u wilt ontvangen).

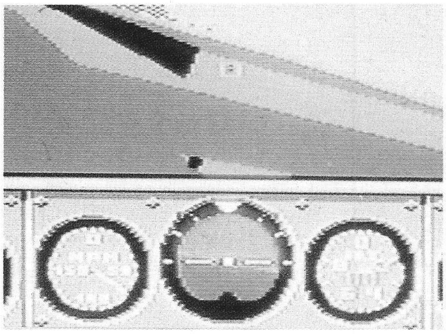
Natuurlijk kunt u voor al deze informatie ook naar het programma kijken, dan heeft u dubbel plezier!

De omslag van de eerste spel-diskette



Chuck Yeagers Advanced Flight Simulator

De naam Chuck Yeagers mag dan als vlieg-crack tot de verbeelding der Amerikanen spreken, in Europa zegt ons dat allemaal niet veel. Electronic Arts vond dat het weer eens tijd werd voor een nieuwe Flight simulator en noemde het product meteen maar **Advanced Flightsimulator**. Plak daar nog eens de naam van een vliegende legende, in dit geval Chuck Yeager, aan en je hebt een topper op de Amerikaanse markt.



Op zich heeft deze EA vluchtsimulator veel te bieden. Er is bijvoorbeeld keuze uit meerdere toestellen zoals de Cessna 172 Skyhawk, F-16 Fighting Falcon, F-18 Hornet, P-15 Mustang, de Piper Cherokee, een Sopwith Camel, de Spad XIII, een SR-71, een Spitfire, een X-1 en een X-3 Stiletto. En of dat allemaal nog niet genoeg is kan de huiskamervlieger ook nog eens zijn/haar "leven wagen" met drie experimentele toestellen, de XNL16 Instigator, de XPG-12 Samurai en een XRH-4 MadDog. Wij zouden niet eens weten hoe de meeste van die toestellen er uit zagen, laat staan hoe ze te vliegen. Maar daar heeft de speler natuurlijk Chuck Yeagers voor.
Is er eenmaal voor een bepaald type vliegtuig gekozen dan biedt het beeldscherm een uitgebreid instrumentenpaneel en uitzicht door de cockpitraampjes. Voor en tijdens de start kijkt de speler naar de landingsbaan en tijdens het vliegen naar de horizon en tal van bekende amerikaanse landmarks. Vreemd genoeg lijken al die instrumenten panelen verrassend veel op elkaar. Als leek zou men toch verwachten dat een Sopwith Camel uit 1917 hemelsbreed verschild met een moderne supersonische jet, maar er valt in ieder geval mee te vliegen. De snelheid en manouvreerbaarheid komen daarentegen beter overeen met het echte vliegtuigtype.

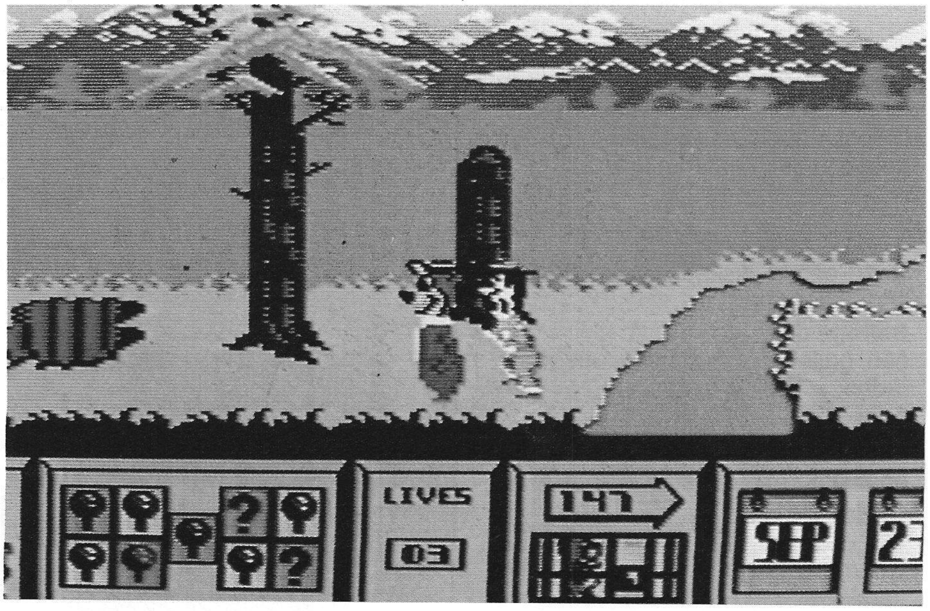
Wat het vliegen betreft heeft de piloot de keuze uit de volgende opties:

- **Test Flight**, probeer alles uit wat de kist te bieden heeft. Zonder te crashen, want anders wordt Chuck boos.
- **Formation Flight** is moeilijker dan het er uit ziet, want als vlieger moet je het leidende toestel ook in allerlei lastige stunts volgen.
- **Airplane Racing**. De computer levert vijf jakkerende tegen-standers die met gevaar voor het eigen leven allerlei gates en slaloms nemen.
- De **Demonstration Flight** leert de testpiloot alles over de acrobatische mogelijkheden van de XPG-12 Samurai, waarbij deze laatste Japanse term te denken geeft. Mischien een vergissinkje met Kamikaze i.p.v. de nobele ridder?
- Voor de rustigere vliegers onder ons is er de **Flight Instruction**, een serie oefeningen in beschaafd vliegen. Desgewenst doet Chuck Yeager het eventjes voor.

Chuck Yeagers AFS is beschikbaar voor de C-64 en IBM PC. Beide versies lijken veel op de wereld-beroemde vluchtsimulator van Sublogic. De verschillen ziten hem in de graphics (betere kleuren op de C-64) en snelheid (de PC is over het geheel sneller in het vliegen).
Een aardig boeiende en goed verzorgd vluchtsimulator die nauwelijks iets toevoegt aan de bestaande FS-cracks. De diskette- versie kost voor de C-64 circa *f*, 35,- en voor de PC circa *f* 90,-.

Yogi Bear

De wat ouderen onder ons zullen de **Yogi Bear**-tekenfilms nog wel herinneren. Yogi, little Boo-Boo en de boswachter Ranger Smith werden wel niet zo beroemd als de Flintstones, maar deden het best aardig. Het softwarehuis zag er wat in om deze drie kartoonfiguren tot een C-64/128 videogame om te werken.
De liefhebber zal zich in de Yogi-, Boo-Boo- en Smith-sprites wel kunnen vinden. Deze sprites lijken veel op de oorspronkelijke tekenstijl van Hanna en Barbara. Het probleem bij dit soort kartoon-omzettingen blijft helaas altijd de sfeersetting. De oorspronkelijke strip- of tekenfilmsfeer evenaren bleek voor menig softwarehuis een te grote klus. Daarvan getuigen de talloze flops in dit genre.
Of u Yogi Bear leuk zal vinden hangt sterk van de persoonlijke smaak en spelbehoeften af. Voor de liefhebber van de oorspronkelijke tekenfilm en strips zijn alle ingrediënten voor een echt Yogi-avontuur aanwezig: De kleine Boo-Boo is door een jager gevangen genomen en Yogi moet hem redden. Er staan tal van onbeheerde picknicktafels in het Yellowstone Park te wachten op plundering door beren. Ranger Smith ligt op de loer om Yogi in de kladden te vatten. Verder zijn er nog wat wilde slangen, heetgebakerde jagers met plenti ammunitie, rivieren, en gemene elanden.
Het competitie-element zit hem in de beschikbare tijd. Yogi is aan zijn winterslaap toe en voor die tijd dient Boo Boo gered te worden. Dat betekent rennen en razen terwijl onderweg kleverige toffe-appels op mogelijke sleutes onderzocht worden.



Tweederde van het spelscherm toont het Ys ellowstone Park en daaronder een info-scherm met de datum, het aantal punten, het aantal levens, de afstand tot Boo Boo en de energievoorraad. Yogi wordt met de joystick naar links/rechts en op/neer bestuurd. Ontsnappen aan de op berehuid beluste tegenstanders kan door snel in een bosje te veranderen. De score betaalt uit overlevings- en afstands (tot Boo Boo)-punten.
Voor de liefhebbers van het vrije bereleven. De cassette-uitvoering gaat rond de *f* 35,- en de disketteversie rond de *f* 50,- kosten.

Lurking Horror

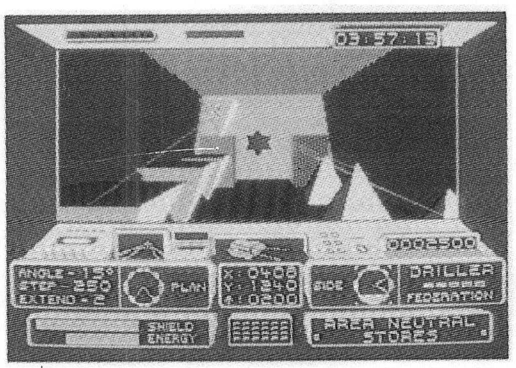
Griezel-software is zeldzaam voor de Amiga en Commodore-64. De meeste pakketten zijn eerder om te lachen of (nog erger) te huilen van ellende. David Leblings (Infocom en co-auteur van Zork)) **Lurking Horror** pretendeert wat meer te zijn dan het doorsnee griezel-videogame. Het verhaal zit doordacht in elkaar en kent interessante verborgen psychotische trekjes. Helaas blijft het niet overal griezelig.
De "nachtmerrie" start als u, in het spel een argeloze student op GUE Techn., het in zijn/haar stomme hoofd haalt om tijdens een verschrikkelijke sneeuwstorm de campus op te gaan om op de computer nog even het werk af te maken. Deetman zou trots zijn op zulke studenten, maar er komt in dit game wel de nodige ellende van. De storm neemt toe en het is niet meer mogelijk om het computer centrum te verlaten. En dan gaan de enge dingen gebeuren.



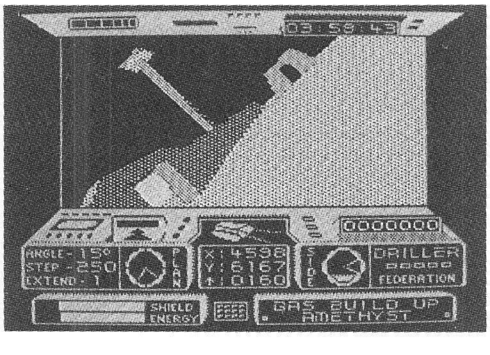
Zoals gebruikelijk in dit soort horrorsettings blijkt men niet alleen in het gebouw te zitten. Eigenlijk verwachtten wij hier zoiets flauws als de dertiende variant van vrijdag de dertiende of een moordende diskdrive, maar verrassend genoeg blijkt het computer-centrum ineens een toegang tot een andere wereld, een specialiteitje van Infocom. Zo wordt de speler die rustig achter een terminal zijn studiedriften botviert plotseling naar een primitieve wereld vol van magie, offers, stamrituelen en een collectie buitenaardse griezels versleept. Dat is weer eens wat anders dan al die duffe data. Het uit(terug)breken uit deze vreemde wereld is voor een handig avonturier of briljante student niet moeilijk. Maar dan rijzen nieuwe problemen: Wie is de geheimzinnige hacker die de toegang tot deze vreemde wereld opende? Wat is op de campus verborgen? En vooral wat loert er achter de deur van het chemische laboratorium? Wie weet wel een uit de hand gelopen onderzoekje met recombinant-DNA of iets dergelijks.
Ontwerper Lebling weet de spanning er best in te houden. Soms wordt de spookachtige entourage even onderbroken met een komische noot. Het aantal puzzels weet in ieder geval uren te boeien.
Ons inziens scoort de Amiga-versie (circa 100 piek) dankzij de geluidsef-fecten wat hoger dan de C-64/C-128-uitvoering. Deze laatste diskette is als troost ongeveer *f* 20,- goedkoper. Een aanrader voor Infocom- en griezelfans.

Boren in 3D

Incentive Software's **Driller** mag grafisch gezien een klein meesterwerkje worden genoemd. Een dergelijk staaltje 3D-kwaliteit vindt men zelden op de C-64. Het Driller-pakket is ook uiterlijk goed verzorgd. Een uitgebreid handboek met begeleidende novelle en 3D-kaart maken de speler direkt thuis in de wereld van het ruimte-boren.
Het verhaal is als volgt: De planeet Evath bezit twee manen: Tricusip en Mitral. Op Mitral waren in het verleden al uitgebreide mijnactiviteiten met het doel om Rubicon energiekristallen te vergaren. Helaas was de oorspronkelijke bevolking, de Ketars, daarin niet zo technisch geschoold en door gasophopingen staat de maan min of meer op springen. De Ketars hebben daarom de maan reeds lang geleden verlaten.
Die maan gewoon lekker laten hangen zult u zeggen. Inderdaad een goed idee ware het niet dat er een komeet komt aanstormen. Een voltreffer zou een voor de planeet Evath rampzalige explosie kunnen veroorzaken.



Uw taak als mijningenieur is alle 18 mijngebieden op Mitral van gas te zuiveren. Daarvoor moet er geboord en het opstijgende gas verbrand worden. De boorman/vrouw werkt vanuit een soort tank. Deze tank is uitgerust met een laser en een teleporteur voor boorapparatuur. Om het moeilijk te maken is uw voertuig gulzig met energie en zullen voortdurend Rubiconkristallen geladen moeten worden. Anders geeft dit voertuig er de brui aan en valt het defensieve energiescherm weg.
Driller beschikt over zeer goede graphics. De speler kijkt door een venster naar het boorgebied en onder dit venster zit het 3D- bedienings- en informatiepaneel van de tank. Menige simulator kan een voorbeeld nemen aan de zorg die de ontwerpers van Driller in dit instrumentenpaneel staken.

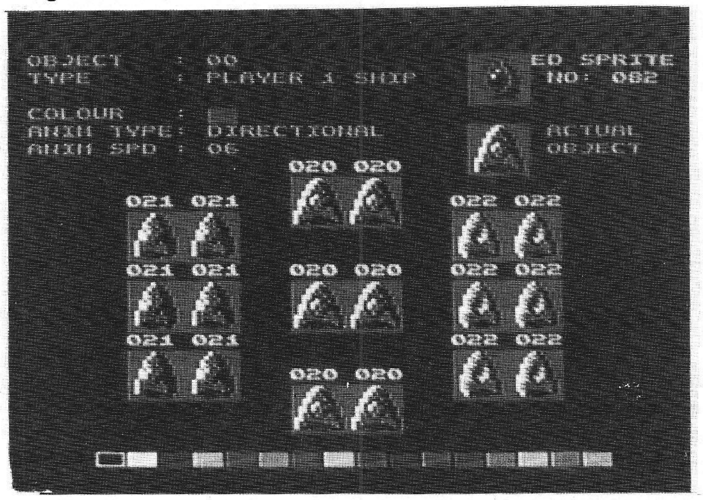


Er zijn meerdere acties mogelijk: De planeet verkennen en boren. Het verkennen is intrigerender dan boren. De Ketars lieten tal van raadsels en onbekende verdedigingsmechanismen achter. In het landschap staan kubussen die als een soort schakelaar naar het onbekende functioneren. Schiet er op met de laser en er opent zich een geheime deur of lift. Verder is het verstandig om de maan eerst in kaart te brengen. Dat gaat gemakkelijker met een verlaten vliegtuig dan met de eigen tank.
Driller komt binnenkort in de winkel voor de C-64 en zal op diskette rond de *f* 75,- gaan kosten.

Nieuws

Shoot them up

Nog niet genoeg space invaders en andere tegenstander naar het nirwana geknald of alle games reeds gespeeld? Maak dan uw eigen Shoot Them Up-game met de gelijknamige construction-kit. De ontwerpers Christopher Yates en Jonathan Hare bieden de gebruiker een veelzijdige sprite-generator, achtergrond-painten, animatie- & botsings-editor en geluidseffecten.



Een krachtig pakket voor het zelf schrijven van onderhoudende software. De C-64 diskette van de Shoot Them Up Construction Kit kost met 4 kant-en-klare demonstratiespelen circa *f* 65,-.

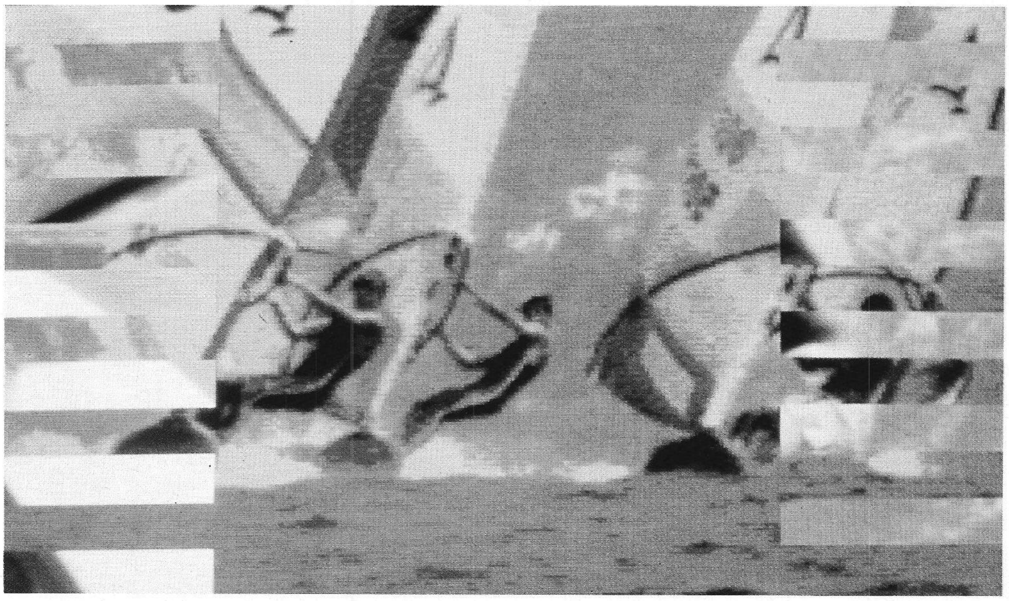
Jinxter

De naam **Jinxter** duidt op iets heksachtigs en dat klopt. In het land Aquitania is hekserij aan de orde van de dag. Geheel heksen eigen werd dat eerst een puinhoop van boosaardige uitspattingen, maar gelukkig wist de goede tovenaar Turani de bevolking met een magische armband te beschermen. Het beheksen en verongelukken werd daarmee meteen de kop in gedrukt. De zwarte magie moest het tegen de groene toverkracht afleggen.
De heks Jannedor wil de goede oude tijden herstellen en slaagt er in de beschermende armband te demonteren en de stukken overal rond te spreiden. Meteen is het zwarte magie-hek weer van de dam en het enige wat de bewoners van Aquitania nog kan redden is een speurtocht naar de onderdelen van de toverarmband.
Deze speurtocht ontwikkelt zich op de Amiga tot een ongewoon en grafisch sterk avontuur. Sterven blijkt niet mogelijk, want het geluk weet de held(in) telkenmale nog net uit alle levensgevaarlijke situaties te redden. Dat lijkt eerst wat saai. In de loop van het spel blijkt echter dat het nemen van extreme risico's ook heel uitdagend kan zijn. Overigens kan het geluk wel degelijk opraken!
Het tweede boeiende spelelement is het oplossen van de gebruikelijke puzzels. Daarvoor heeft u geen geluk, maar juist vindingrijkheid nodig. Tot slot de allom vertegenwoordigde humor die de speler zelfs in de moeilijkste situaties op de been weet te houden.
Een van de betere Amiga adventures met een krachtige parser (die gecompliceerde opdrachten begrijpt) en 30 topkwaliteit spelschermen. De prijs van dit Engelstalige spel bedraagt circa *f* 90,-.

Zelf mooie computertekeningen maken, en vervolgens die tekeningen in zelfgemaakte programma's invoeren, is geen makkelijke opgave. Daar komt heel wat programmeerwerk bij kijken. Het volgende artikel kan daar echter verandering in brengen.

Conversie van tekeningen

Koala Painter en Cheese als tekenhulp



Gelukkig zijn er voor de verschillende Commodores een aantal goede tekenprogramma's te koop, waarvan Koala Painter en Cheese (die met de muis werkt) de meest bekende zijn. Er wordt echter bij de programma's niet vermeld hoe je je tekeningen in kan bouwen in eigengemaakte programma's. Vandaardeze speurtocht naar een methode om Koala Painter en Cheese als tekenhulp te gebruiken.

Vroeg of laat krijgt iedereen, die zelf programmeeert wel eens de behoefte om tekeningen en grafische voorstellingen toe te voegen. Meestal een heel gedoe, als je dat ook nog moet programmeren. De mogelijkheid om daarvoor andere - reeds bestaande - programma's voor te kunnen gebruiken, is voor veel doe-het-zelf programmeurs een welkome uitbreiding.

Onderdelen

Wanneer je je fraai ontworpen tekening gesaved hebt (geen overbodige luxe) is het logisch dat het gesavede datablok minstens de volgende onderdelen moet bevatten:

° De 8K grote BitMap-data (de eigenlijke tekeningen).

° 1K aan kleurendata die in het gedeelte van adres 1024 tot 2023 komt te staan.

° 1K aan kleurendata voor gedeelte 55296 tot 56295.

Het volgende wat je dan moet doen is de bovengenoemde onderdelen op te zoeken in je tekening-file.

Koala Painter (disk versie).

Ik laat nu zien hoe dat bij door Koala Painter gemaakte tekeningen gaat, maar in principe is deze methode voor elk tekenprogramma hetzelfde.

Het eerste wat ik gedaan heb is het beginadres van de tekening- file te noteren. Het eindadres noteren is ook handig, maar is geen vereiste. Je kan daarvoor gebruik maken van diverse Disk- hulpprogramma's, maar het volgende BASIC-programmaatje (voor het beginadres) werkt ook:

```
10 OPEN 1,8,0,"NAAM"
20 GET#1,A$,B$
30 IF A$=""THEN A$=CHR$(O)
40 IF BS=""THEN B$=CHR$(O)
50 PRINT ASC9A$)+256*ASC(B$)
60 CLOSE 1
```

Het begin- en eindadres van een tekening-file was resp. 24576 en 32576. De drie hierboven besproken onderdelen bevatten samen 8000 (tekening) + 1000 (kleur) + 1000 (kleur) = 10000 bytes en dat is precies het aantal bytes in onze tekening-file dus dat maakt onze speurtocht al wat makkelijker..

Voor we onze speurtocht beginnen moeten we eerst onze tekening- file in het geheugen laden. Het is een gewoon datablok dus laden we met "8,1" en geven we het NEW-commando om alles te resetten.
Bij Koala Painter bevind zich een raar 'hartje'-teken aan het begin van de file-naam en om het ons makkelijker te maken laden we de file als volgt:

LOAD "?NAAM*,8,1

Het datablok zit nu in het geheugen en we kunnen onze speurtocht starten.

Op zoek

Het begin van de BitMap-data ligt meestal op een 'makkelijk' geheugenadres. Meestal is hiervan de LowByte nul. Laten we daarom maar met geheugenplaats 24576 ($6000) beginnen door de hieropvolgende 8000 bytes kopieren in het, door ons beter bruikbare, gedeelte vanaf adres 8192.
Met wat moeite kunnen we onze tekening ontdekken. Door Multi Colour Mode in te schakelen wordt het al helemaal duidelijk. Alleen de kleuren blijken anders te zijn.
Ook nu wordt het weer uitproberen door blokken 1000 bytes in de gedeelten van 1024-2023 en 55296-55295 te kopieren (vergeet niet dat de bytes in 102402023 uit 2 kleuren in 1 bestaat: De HighByte voor BitPaar '01' en de LowByte voor BitPaar '10').
Voor onze tekening-file blijven er maar 2 blokken van 1000 bytes over dus je hebt 50% kans dat je het goede blok in het juiste bereik kopieert.
Na wat uitgeproberen heb ik de volgende MemoryMap samengesteld:

En verder...

Nu je weet waar wat ligt kun je nu je tekening- en kleurenbytes plaatsen en saven waar je wilt en ze in je eigen programma's inbouwen. Om ze in ieder geval op het scherm te laten verschijnen kan het volgende BASIC-programmaatje van dienst zijn:

```
1  POKE 53281,1
10 POKE 53272, PEEK (53272) OR
   8
20 POKE 53265, PEEK (53265) OR
   32
30 POKE 53270, PEEK (53270) OR
   16
40 FOR 1=O TO 7999: POKE 8192
       +1,PEEK(24576+1):NEXT I
50 FOR 1=O TO  999: POKE 1024
       +1,PEEK(32576+1):NEXT I
60 FOR 1=O TO  999: POKE 55296
       +1,PEEK(33576+1):NEXT I
```

Cheese (cassette-versie).

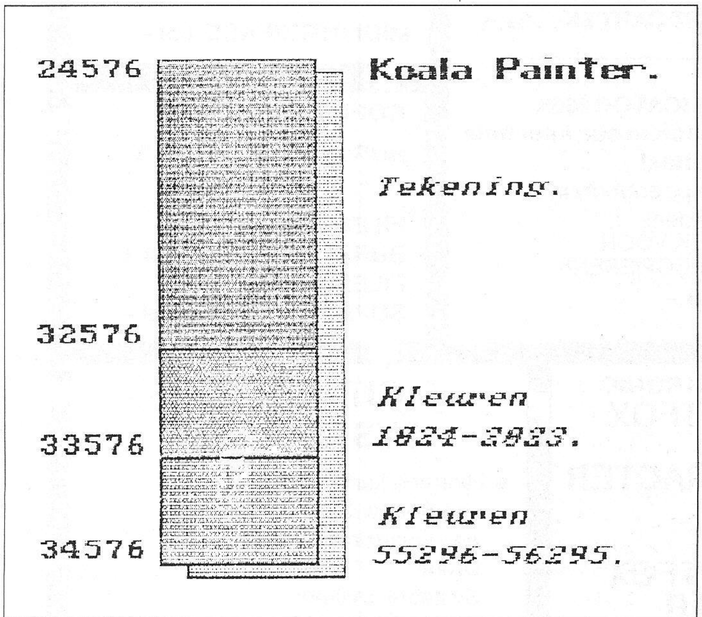
Bij Cheese, het programma met de muis, was het allemaal wat moeilijker. Bij dit programma is het namelijk mogelijk tekeningen te maken die groter zijn dan het scherm. Met veel pijn en moeite is het mij toch gelukt de diverse onderdelen op te sporen zoals te zien is in de volgende MemoryMap (zie afb.2).
Nu hebben we meestal het bovenste gedeelte van de Cheese-tekening nodig. Het volgende BASIC-programmaatje laat zien hoe je dat bovenste gedeelte op je scherm kan laten verschijnen.

```
10 POKE 53281,1
20 POKE 53272,PEEK(53272) OR 8
30 POKE 532265,PEEK(53265) OR
   32
40 POKE 53270,PEEK (53270) OR
   16
50 FOR 1=O TO 7999:POKE 8192
       +1,PEEK(32768+1):NEXT I
60 FOR 1=O TO  999:POKE 1024
       +1,PEEK(29184+1):NEXT I
70 FOR 1=O TO  999: POKE
       55296+1, PEEK(30720
       +1):NEXT I
```

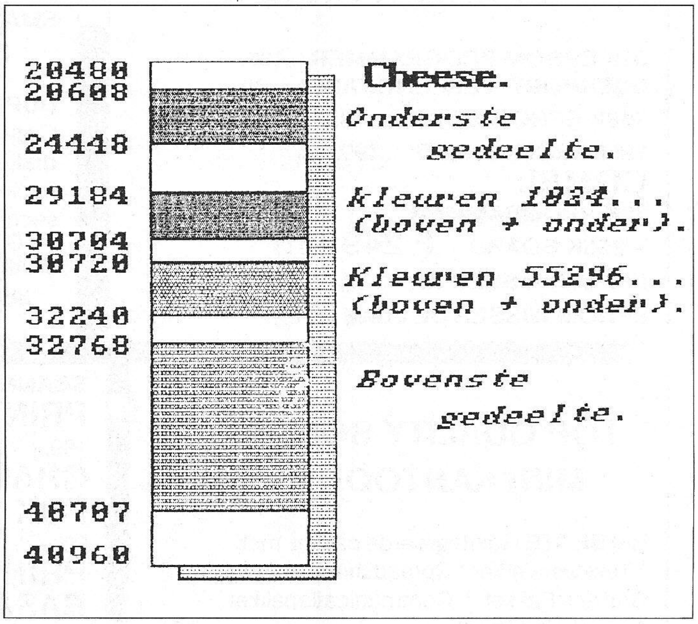
Vergeet vooral niet voordat je het programma runt, eerst de tekening-file in het geheugen te laden. Dat is wel zo handig..

The beat goes on.

Deze BASIC-programma's hebben natuurlijk wel één nadeel: ze zijn een beetje traag. Vandaar dat het wel handig zou zijn om er machinetaal-versies van te maken.
Wat je ook zou kunnen doen is van een Koala-tekening een Cheese tekening te maken (of andersom). Het is allen een kwestie van de juiste onderdelen in de goede geheugenblokken te plaatsen en op de goede manier te saven. Zo kan je ondermeer een Koala-tekening met Cheese verder verwerken.
Je hebt natuurlijk niet altijd de hele tekening nodig. Je zou bijvoorbeeld een 'splitscreen'-interrupt op het scherm kunnen plaatsen.
Zo zijn er nog veel meer mogelijkheden, die je vanzelf ontdekt als je eenmaal een beetje bedreven bent geworden met dit 'vertalen'.



Afb.1: MemoryMap voor Koala Painter



Afb.1: MemoryMap voor Koala Painter

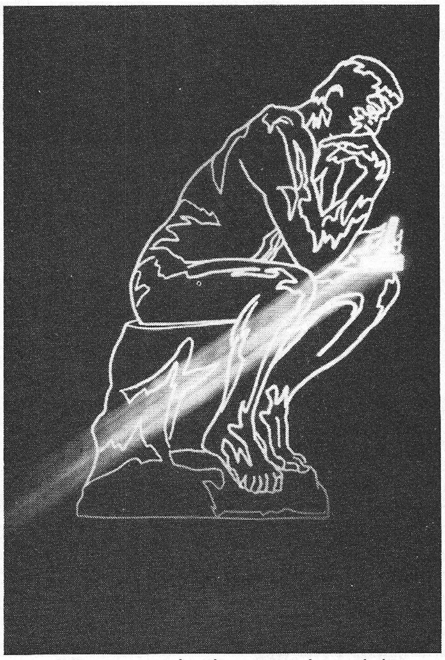
In deze aflevering van onze serie bestemd voor zelfprogrammerende leerkrachten gaat Bob Munniksma in op wat meer machinegerichte zaken zoals het maken van sprites en joystickbesturing van die sprites. Deze onderwerpen krijgen gestalte in een leerzaam memoryspel voor de Commodore 64.

Zelf leerprogramma's maken in BASIC

Deel 8 : Woorden memory

Jonge kinderen, die bezig zijn met het leren lezen, hebben baat bij veel oefening en het veel zien van de globaal aangeboden woorden uit de methode voor leren lezen. Vastleggen van het woordbeeld kan op veel manieren verwezenlijkt worden. De leesmethode zelf verschaft vaak meer dan voldoende oefenstof. Een school, die het onderwijs wil ondersteunen met computers, kan met dit programma-voorbeeld zijn voordeel doen.

Als kinderen gaan leren lezen, dan is het in eerste instantie belangrijk om de woorden waar het om gaat, globaal te leren herkennen. Dat gaat in elk geval op voor een globaalmethode voor aanvankelijk leesonderwijs. Het woord als geheel heeft de nadruk. In een later stadium zullen de kinderen de klankbepalende delen, de letters dus, leren herkennen en verklanken. De woorden, die deel uitmaken van de methodiek, zullen dus veelvuldig en op veel verschillende manieren aan de orde komen. In speelleesmateriaal, werkboekjes, werkbladen en talloze andere leermiddelen zullen de woorden die in een bepaald lespakket centraal staan eindeloos worden ingeslepen, zodat de kinderen de woorden gaan herkennen als horende bij een bepaalde visuele voorstelling. In de methode voor aanvankelijk leesonderwijs van uitgever Zwijssen is dat ook het geval. Deze methode is algemeen in gebruik in Nederland. U zult de woorden wel herkennen. De eerste leskern bestaat uit de woorden: boom, roos, vis, vuur, mus en pim, die in de vorm van een sprookje dagelijks worden aangeboden. Bij het ontwerpen van dit memory-spel ben ik uitgegaan van de globaalwoorden uit deze methode. Ieder kan overigens vrijelijk de gebruikte woorden vervangen door andere. Ik kom hier nog wel op terug. Oefenen en nog eens oefenen is het devies en waarom zou de computer geen rol kunnen spelen in het leerproces. De uitgever van de genoemde leesmethode heeft zelf uitgebreide programmatuur ontwikkeld ter ondersteuning van het leren lezen. Deze software is gemaakt voor de MSX-2 computer met kleurenscherm. Maar zelf kunt u ook behoorlijk zinnig aan de slag. De aflevering van deze maand zal weer bewijzen, dat leerkrachten, die een beetje kunnen programmeren, zelf de nodige software kunnen ontwikkelen, heel gericht aangepast aan de behoeften. Het programma-voorbeeld heeft daarbij de functie van startschot, brainstormer.



Memory: denker met joystick

Memory

Wie kent niet het memory-spel. In vele vormen en maten is dit spel verkrijgbaar. Van een versie voor de allerkleinste, met eenvoudige voorstellingen tot de wat ingewikkelder. Een kind kan al snel meedoen, want de spelregels zijn eenvoudig. Het enig noodzakelijke attribuut is een goed geheugen. In de praktijk zal steeds weer blijken, dat jonge kinderen het met gemak kunnen winnen van volwassenen, eenvoudigweg omdat hun kortdurend geheugen beter is. Memory traint het geheugen en de visuele waarneming. Een memory-spel kan dus een zinvolle functie krijgen bij het automatiseren van het woordbeeld van de globaalwoorden die worden gehanteerd bij een aantal leesmethoden.
Nu kunnen we alle woorden op kaartjes schrijven en zo zelf een memory maken. Het zelf maken van leermiddelen, vooral in de spelsfeer, is in het Basisonderwijs geen uitzonderlijke gebeurtenis. Zo'n zelfgemaakt spel maakt het leren weer wat afwisselender. Dus waarom niet een memory op het beeldscherm gezet. We weten inmiddels wat voor magische werking een beeldscherm op mensen heeft.

Analyse

Alvorens een computer-memory te kunnen ontwerpen, moet eerst het spel zelf worden geanalyseerd. Alleen dan kan men een simulatie van het spel ontwerpen. Bij memory gaat die analyse echter niet zo diep. Bij elke beurt mogen er twee kaartjes worden omgedraaid. Zijn ze niet gelijk, dan moeten ze weer gedekt worden teruggelegd op dezelfde plaats. De kaartjes worden van tevoren geschud en tevens wordt er bepaald, met hoeveel kaartjes er gespeeld wordt. Niet alle kaartjes hoeven gebruikt te worden. Hele jonge kinderen kunnen al meespelen als men bijvoorbeeld de helft van het spel gebruikt. Het verloop van het spel is zo voor het jonge kind ook nog overzichtelijk.

Vervolgens moet de analyse vertaald worden naar computerbegrippen en er moeten bepaalde keuzen worden gedaan, alvorens met het ontwerpen te beginnen. In dit ontwerp ben ik uitgegaan van één speler, maximaal 40 kaartjes (20 woorden dus) en het spelen met behulp van een sprite gestuurd door een joystick in poort 2. Nu geeft het gebruik van sprites een extra dimensie aan een zelfgemaakt spel. Tevens worden de mogelijkheden van de Commodore 64 wat meer benut.
De idee achter het spel is de volgende:
Uitgaande van de algemene spelregels van het memory-spel kan de speler door middel van een joystick een kaartje op het beeldscherm aanwijzen. Door op de vuurknop van de joystick te drukken worden het kaartje als het ware omgedraaid en het woord wordt zichtbaar. Op een gelijke wijze wordt een tweede kaartje omgedraaid en kan er worden gekeken of de woorden gelijk aan elkaar zijn. Zoniet, worden de kaartjes weer gesloten op het beeldscherm teruggeplaatst. Zijn de woorden wel gelijk, dan blijven de kaartjes open op het scherm staan. Ze doen in het spel verder niet meer mee.
De speler kan zelf bepalen hoeveel kaartjes op het scherm moeten komen, aangegeven in veelvouden van acht. Zo heeft het spel vijf spelniveau's zodat ook de wat minder sterke leerlingen mee kunnen doen met het spel. Afhankelijk van het gekozen aantal kaartjes krijgt de speler krediet. Bij elke misser wordt het krediet verminderd. Als alle krediet op is en niet alle kaartjes zijn gevonden, eindigt het spel verloren voor de speler. Vindt hij of zij alle woorden wel voordat het krediet op is, dan wint de speler. In beide gevallen mag nog een keer worden gespeeld.

? ? ? ?

De presentatie

De schermpresentatie van een dergelijk spel moet toch wel enigszins aantrekkelijk zijn voor de speler. Dat is heel goed mogelijk, alhoewel er slechts in geringe mate grafisch geprogrammeerd wordt. Alleen de sprite, in de vorm van een groot vraagteken, en de animatie ervan vereisen wat meer programmeertechniek. Verder is alles zeer eenvoudig BASIC.
Op het scherm liggen maximaal veertig kaartjes in rijen van acht. Onder die kaartjes liggen 20 paren woorden verscholen. Door het vraagteken op een bepaald kaartje te brengen en vervolgens op de vuurknop van de joystick te drukken, kan de speler het kaartje omdraaien en bekijken. Onder aan het scherm staat een lange balk ter grootte van het krediet van de speler. Elke misser verkleint het krediet en de balk. Of de speler nu meer of minder kaartjes legt, het krediet is evenredig.

Programmeerproblemen

Bij het nader bekijken van de analyse ontdekt de ontwerper een aantal zaken, die in het programma zelf verwerkt zullen moeten worden. Zo kan het gebeuren, dat de speler het vraagteken op een reeds geopend kaartje plaatst. Het programma zal dus een reeds geopend kaartje moeten kunnen herkennen.
Dan kan normaal gesproken een sprite niet zonder meer over het hele scherm bewegen. Aangekomen op ongeveer twee-derde van de rechter schermhelft, kan hij niet verder. Er zullen dan aanvullende grootheden geprogrammeerd moeten worden om de sprite toch soepel te laten overstappen tot het rechter schermgedeelte. Vervolgens moet de joystick worden

uitgelezen. Hiervoor zijn vele manieren. Ik koos voor een duidelijke manier om de bewegingen van de joystick om te zetten in bewegingen van het vraagteken. Wel wat langzaam, maar heel soepel. En snelheid heeft in dit spel niet de hoogste prioriteit.
De computer moet ook de positie van de twee omgekeerde kaartjes tijdelijk "onthouden" om ze later weer te kunnen "omdraaien". Tenslotte is het leuk als de kaartjes niet steeds op de zelfde plaats terecht komen. Schudden voor het gebruik maakt het spel steeds weer opnieuw tot een uitdaging.
Hoe al deze gegevens zijn verwerkt in het programma zal ik u vervolgens aan de hand van de programmalisting duidelijk maken.

De listing

Als we de listing bekijken, dan vallen de afzonderlijke delen op door een lege regel met een dubbele punt. Het is een goede gewoonte om functieblokken binnen een programma visueel los van elkaar te plaatsen. Dat maakt de listing leesbaar. Eveneens leesbaar wordt het programma door het feit, dat de regels niet veel langer zijn dan veertig karakters.


Een slechte verliezer ??!

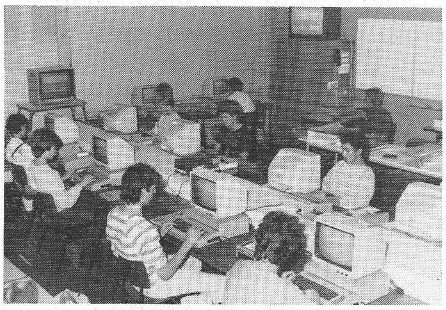
In de regels 100 - 170 worden een aantal zaken voorbereid. Dimensionering van de gebruikte array's en het vullen van strings zullen zeker geen problemen geven. NE$ is weer de string voor het positioneren van de daarna volgende PRINT-opdracht. KO$ geeft een open kaartje op het scherm en KK$ een gesloten kaartje van drie bij vier cursorblokjes groot. Daarna worden de data voor de sprite ingelezen en in een geheugenblok gezet (vanaf regel 180). Hoe een sprite precies gemaakt wordt kun u later lezen. Dan worden de gebruikte woorden in tweetallen in array WW$(X) gezet (regels 205 - 220).
Dan volgt op regel 230 de vraag aan de speler met hoeveel rijen hij of zij wil spelen. Geen RETURN-indruk nodig, want de computer wacht op een enkele toetsindruk. Vervolgens worden het aantal gekozen kaartjes door elkaar geschud en tevens wordt er de string SC$ voor het krediet gemaakt. Let even op het stukje programma in de regels 285 - 310. Daar worden de woorden gekoppeld aan de kaartjes op het scherm. Als er een kaartje is "neergelegd" dan kan er nadien op die positie geen meer worden neergelegd. Dat is natuurlijk heel vanzelfsprekend. Deze programmeertruc is trouwens ook erg handig bij andere leerprogramma's waarbij een aantal random getallen worden gegenereerd, zonder dat er een getal dubbel voor mag komen. Het gebeurt wel in tafel- programma's, waarbij de tafels willekeurig door de computer worden

bedacht, zonder dat er vlak na elkaar dezelfde opdracht verschijnt. Door elk opgewekt getal te bewaren in een aparte array, is het mogelijk een random getal te toetsen aan reeds eerder gegenereerde getallen. Als het getal reeds is geweest, dan kan het programma doorrollen tot er een nog niet verschenen getal wordt opgewekt.

Als alle kaartjes geschud en gepositioneerd zijn kan het scherm worden gevuld met de dichte kaartjes. Dat gebeurt in de regels 325 - 350.

Het spel zelf begint op regel 360. Het stuurgedeelte van het spel loopt niet verder dan regel 420. Voor de rest wordt alles gerealiseerd door middel van sub-routines, die vanuit het stuurgedeelte (360-420) al naar gelang de voorwaarden worden aangeroepen. Zo is er voor elke handeling een subroutine. Deze werkwijze maakt het aanpassen van het programma aan specifieke eisen erg eenvoudig.

Laten we het spelverloop eens nader bekijken. Omdat het stuurgedeelte na elke beurt steeds weer helemaal wordt afgewerkt, kijkt het programma eerst of er aan voorwaarden wordt voldaan, die het spel laten eindigen. Het krediet is op of alle woorden zijn gevonden. Als ME=4 of ME=5 dan eindigt het spel. Is dat niet zo, dan wordt de sprite voor de volgende (in dit geval de eerste) beurt aangezet met POKE S+21,3. Raadpleeg uw handboek of referentiegids voor nadere details.



Vervolgens gaat het programma naar de uitlees-routine van de joystick. Ik heb gekozen voor de joystick op poort 2. Dat is het eenvoudigst te programmeren. Vanaf regel 655 worden de bewegingen van de joystick en de vuurknop omgezet in waarden, die vervolgens op hun beurt weer verwijzen naar een sub-routine, die de sprite-animatie bewerkstelligen. Deze sub-routine draait in een dichte programmalus tot de speler op de vuurknop drukt (regel 670). In de joystick-routine wordt behalve de positie van de sprite ook de relatie tot het te openen kaartje bijgehouden in regel 695. In de variabele K staat steeds het nummer van de te openen kaart. Als de speler op de vuurknop drukt, wordt K gefixeerd en het programma springt terug naar het stuurgedeelte. Dan wordt de geldigheid gecontroleerd en teruggegaan naar de keuze van het eerste kaartje als de kaart reeds geopend was. Is de kaart wel geldig, dan wordt deze omgedraaid (regel 525 - 535) en mag de speler de tweede kaart uitzoeken, weer door middel van de joystick-routine. De tweede kaart wordt eveneens op geldigheid gecontroleerd en de coördinaten van de sprite-positie worden vanaf regel 545 omgerekend naar het nummer van het array-element en de positie van het bijbehorende kaartje op het scherm.

Afhankelijk van de omgekeerde kaartjes zal het programma een aantal handelingen verrichten. Als de woorden op de kaartjes niet identiek zijn, dan worden de kaartjes na twee seconden (regel 500) weer terugge-

```
10 rem ******************************
20 rem **       woorden memory     **
30 rem **                          **
40 rem ** leerprogramma's in basic **
50 rem **                          **
60 rem ** (c)1988   bob munniksma  **
70 rem **                          **
80 rem **   voor commodore info    **
90 rem ******************************
100 :
105 rem initialisatie
110 dim w$(40),ww$(40),ka$(20),r(40)
115 print chr$(14)chr$(8)chr$(147);
120 poke 53280,0:poke 53281,0
125 ne$=chr$(19):for i=1 to 23
130 ne$=ne$+chr$(17):next
135 :
140 k2$=chr$(17):k3$=chr$(18)
145 for i=1 to 4:k1$=k1$+chr$(169)
150 k2$=k2$+chr$(157)
155 k3$=k3$+chr$(32):next
160 ko$=chr$(158)+k3$+k2$+k3$+k2$+k3$
165 kk$=chr$(159)+k1$+k2$+k1$+k2$+k1$
170 px=24:y=0:sg=0
175 :
180 rem sprite inlezen
185 s=53248:poke 2040,13
190 for i=0 to 62:read d
195 poke 13*64+i,d:next
200 :
205 rem array's inlezen
210 for i=1 to 40 step 2:read ww$(i)
215 ww$(i+1)=ww$(i):next
220 for i=0 to 5:read me$(i):next
225 :
230 rem instellen aantal rijen
235 print chr$(147)"hoeveel rijen?(1-5)"
240 get t$:if t$="" then 240
245 ak=val(t$):if ak<1 or ak>5 then 240
250 ak=8*ak
255 :
260 rem kaarten schudden
265 me=0:sc=ak:g=0
270 for i=1 to ak:sc$=sc$+chr$(162)
275 sa$=sa$+chr$(32):next
280 print chr$(147)"    schudden"
285 for i=1 to 20:ka$(i)=""
290 r(i)=0:r(i+20)=0:next:r=0
295 q=int(rnd(1)*ak)+1:for i=1 to ak
300 if r(i)=q then 295
305 next:r=r+1:r(r)=q:print chr$(19)r
310 if r<ak then 295
315 for i=1 to ak:w$(i)=ww$(r(i)):next
320 :
325 rem scherm vullen
330 print chr$(19)chr$(18);
335 print"       WOORDEN- M E M O R Y     "
340 for j=1 to 8:print chr$(19):print
345 for i=1 to ak/8:print tab(j*5-5)kk$
350 print:next i,j
355 :
360 rem spel gedeelte
  365 gosub 570:if me=4 or me=5 then 625
  370 poke s+21,3:gosub 655:gosub 595
  375 if z=1 then me=1:gosub 570:goto 360
  380 me=0:gosub 570
  385 gosub 525:y1=11:x1=12:w1$=w$(k)
  390 gosub 655:gosub 595
  395 if z=1 then me=1:gosub 570:goto 390
  400 gosub 525:y2=11:x2=12:w2$=w$(k)
  405 if x1=x2 and y1=y2 then gosub 430:goto
  390
  410 poke s+21,0
  415 if w1$=w2$ then gosub 445:goto 360
  420 gosub 470:goto 360
  425 :
  430 rem dezelfde kaart omgedraaid
  435 me=1:gosub 570:return
  440 :
  445 rem goed gekozen
  450 g=g+1:ka$(g)=w1$
  455 if g=ak/2 then me=4:gosub 570:return
  460 me=3:gosub 570:return
  465 :
  470 rem fout gekozen
  475 sc=sc-1
  480 sc$=left$(sa$,(ak-sc))+right$(sc$,sc)
  485 me=2:gosub 570
  490 :
  495 rem foute kaarten sluiten
  500 for i=1 to 2000:next
  505 print left$(ne$,y1)tab(x1)kk$
  510 print left$(ne$,y2)tab(x2)kk$
  515 return
  520 :
  525 rem gekozen woord plaatsen
  530 gosub 545:print ko$:gosub 545
  535 print chr$(17)chr$(18)w$(k):return
  540 :
```

draaid. Het krediet loopt met één terug. Zie daarvoor de regels 470 - 515. Wordt een kaart aangewezen, die al is omgedraaid, dan moet de speler een andere kaart kiezen. Dat ziet u in de regels 430 en 435.

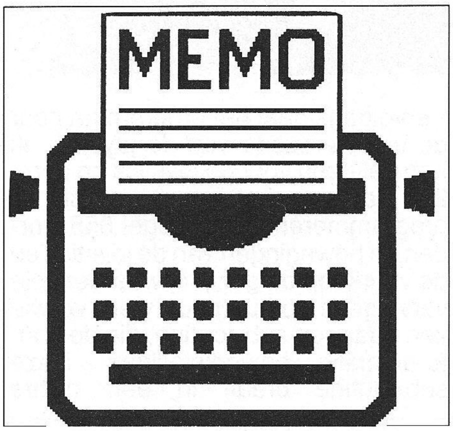
De sprite wordt na het kiezen van beide kaartjes uitgezet, zodat de speler de woorden goed kan bekijken. Is het spel teneinde, dan mag de speler door op de J-toets te drukken, nog eens spelen. U kunt de verdere afloop zelf aan uw wensen aanpassen in de regels 625 - 645. Bij "nee" kunt u bijvoorbeeld een eigen afbuiging maken. Als het antwoord "ja" is, dan zal het opgebouwde krediet worden aangevuld met nieuw krediet. Een speler, die de eerste ronde vlot deed, heeft dus voor een volgende ronde meer beurten te missen. Als u elk spel met evenveel krediet wil laten starten, dan moet u regel 640 veranderen in:

640 if t$="j" then run

Alle bestaande variabelen worden dan op nul gezet.

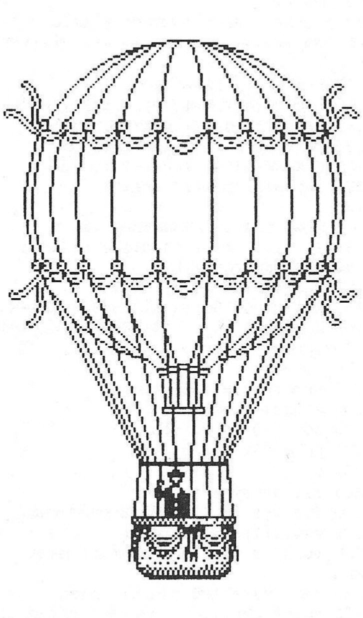
Sprite animatie

Tot slot van deze aflevering iets meer over het programmeren van sprites op de Commodore 64. Er zijn al heel wat boeken en cursussen over gemaakt, maar ik wil u wat praktische zaken toch niet onthouden. De sprite, die in het programma-voorbeeld wordt gebruikt, heeft de vorm van een vraagteken. U kunt de vorm als u dat wenst zelf veranderen, zonder de rest van het programma te hoeven aanpassen. De gegevens van de sprite vindt u in de regels 820 - 925. Ze worden ingelezen met de regels 180 - 195.



Als u een sprite ontwerpt, kunt u daar het beste een zogeheten sprite-generator voor gebruiken. Een hele goede is de Sprite-machine van Topsoft. U kunt de sprite punt voor punt in het groot op het scherm maken en bekijken. Als de vorm naar uw zin is, genereert Sprite-machine vanzelf de DATA-regels in BASIC op tape of diskette. Met wat knip- en plakwerk is de klus dan zo geklaard. Doet u het handmatig, dan zit er wat meer werk aan vast. Raadpleeg steeds uw computerhandboek of een ander goed Commodore 64 boek. De gegevens in DATA komen steeds in triootjes. Eén-entwintig regels van drie getallen.
Om een sprite te maken neemt u bijvoorbeeld ruitjespapier en tekent daarop een raster van 21 regels van 24 hokjes. De 24 hokjes op één regel zijn weer onderverdeeld in clusters van 8. Teken een sprite-ontwerp door de gewenste hokjes in te kleuren. Dan kunt u met het berekenen van de DATA-getallen overgaan. Een wit hokje is een 0 en een gekleurd hokje heeft achtereenvolgens de waarden 128, 64, 32, 16, 8, 4, 2 en 1. Dat doet u zo drie keer op elke regel. Tel van elk cluster van acht de waarden van

```
545 rem lokaliseren
550 l1=int(y/32)*4+3:l2=int(x/40)*5
555 print left$(ne$,l1)tab(12);:return
560 :
565 :
570 rem mededelingen
575 if sc=0 then me=5
580 print ne$;tab(40-ak)sc$;
585 print left$(ne$,23)me$(me);:return
590 :
595 rem kaart controle
600 if sc=0 then return
605 z=0:for i=1 to g
610 if ka$(i)=w$(k) then z=1
615 next i:return
620 :
625 rem einde spel
630 print" nog een keer? J/N"
635 get t$:if t$="" then 635
640 if t$="j" then goto 230
645 goto 635
650 :
655 rem joystick besturing
660 jo=56320:a=peek(jo):v=a and 16
665 a=15-(a and 15)
670 if v=0 then return
675 if a=1 then gosub 720
680 if a=2 then gosub 740
685 if a=4 then gosub 760
690 if a=8 then gosub 780
695 k=1+int(x/40)+int(y/32)*8
700 poke s+16,sg:poke s,px
705 poke s+1,y+66
710 goto 655
715 :
720 rem omhoog
725 y=y-5:if y<0 then y=0
730 return
735 :
740 rem omlaag
745 y=y+5:if y>4*ak-30 then y=4*ak-30
750 return
755 :
760 rem links
765 x=x-5:if x<0 then x=0
770 gosub 800:return
775 :
780 rem rechts
785 x=x+5:if x>290 then x=290
790 gosub 800:return
795 :
800 rem verbreden sprite-gebied
805 xx=x-231:if xx<0 then
sg=0:px=x+24:return
810 sg=1:px=xx:return
815 :
820 rem sprite data
825 data 000,000,000
830 data 001,252,000
835 data 007,255,128
840 data 031,255,192
845 data 063,007,224
850 data 126,003,224
855 data 126,003,224
860 data 060,007,224
865 data 000,015,192
870 data 000,031,128
875 data 000,063,000
880 data 000,126,000
885 data 000,252,000
890 data 001,248,000
895 data 003,240,000
900 data 003,240,000
905 data 003,240,000
910 data 000,000,000
915 data 003,240,000
920 data 003,240,000
925 data 003,240,000
930 :
935 rem woorden
940 data boom,roos,vis,vuur,mus,pim
945 data bel,boek,pet,miep,kees,aap,raam
950 data oom,bus,sam,eet,kar,tol,mee
955 :
960 rem mededelingen
965 data "        "
970 data "nog eens"
975 data "mis     "
980 data "goed    "
985 data "gewonnen ! "
990 data "verloren ! "
```



de ingekleurde hokjes bij elkaar op. Zo krijgt u voor elke regel drie decimale getallen. U doet op dezelfde wijze de overige 20 regels. Als u de 63 getallen heeft berekend, kunt u ze op de bekende wijze verwerken in uw programma. Inlezen en de sprite staat na aanzetten tot uw beschikking. Voor alle volledigheid moet wel worden vermeld, dat het om een één-kleurige sprite gaat. Meer-kleurige sprites geeft nog iets meer werk.
De decimale getallen komen in een leeg geheugendeel terecht volgens de formule in regel 195. U kunt het spriteblok in dit geval toewijzen met POKE 2040,13. Het verplaatsen van de sprite over het rechter schermdeel behoeft zoals gezegd wat meer programmeerwerk. Als u het scherm bekijkt, dan zult u kunnen vaststellen, dat de lengte ruim 300 beeldpuntjes (ook wel pixels) beslaat. De hoogte heeft ongeveer 200 pixels. Die 300 pixels worden nog eens uitgebreid met ongeveer 50 pixels, die er wel zijn, maar zich niet op het beeldrechthoek bevinden. U kunt een sprite zo uit beeld laten verdwijnen. 350 pixels laten zich met 256 (uw computer is immers 8-bits) waarden niet allemaal besturen. Het werkelijke scherm loopt nu van links naar rechts en begint bij 0. Op pixel 24 begint het zichtbare beeldscherm aan de linkerkant op uw monitor of tv. Dat gaat door tot pixel 255. U zult beseffen, dat u dan niet verder bent gekomen dan op circa twee-derde van het scherm. De laatste derde is alleen bestuurbaar als u een register met de waarde 1 vult en met de pixelwaarde opnieuw gaat tellen. Bekijk de programmaregels 800 - 810 en de regel 700 maar eens goed. Op deze wijze wordt een soepele overgang van linkergebied naar rechtergebied en terug bewerkstelligt. U kunt deze joystick-routine voor het besturen van een sprite ook los in bijvoorbeeld een ander of eigen programma gebruiken. U moet dan niet vergeten de variabele S te definiëren. Tot zover deze aflevering. Veel succes met typen en de leerlingen zullen zeker veel plezier beleven aan deze manier van memory spelen.

B.M.

Gebruikersgroepen

Hobbycomputerclub Futura

Voor gebruikers van home- en personalcomputers in de Zaanstreek bestaat er al enkele jaren een vereniging. Deze vereniging richt zich niet op een bepaald merk of soort computers, maar meer op de gebruikers zelf.
Hoe vaak gebeurt het niet dat je via bijvoorbeeld een PC-prive project een computer aanschaft terwijl achteraf blijkt dat het toch iets moeilijker is dat je in eerste instantie dacht? Juist voor dit soort gebruikers is het handig als er in de buurt een vereniging is die wekelijks bijeenkomsten organiseert. Op zo'n bijeenkomst kan je je probelemen aan andere gebruikers voorleggen. Vaak hebben die jouw problemen ook al eens gehad, en waarschijnlijk is er toen ook een oplossing gevonden. Voor de leden die een modem hebben, is het zelfs mogelijk om 24 uur per dag contact te maken met onze databank FUTURA-BBS, waar experts in o.a. dBase-III, Amiga, MS-DOS en ATARI ST je verder kunnen helpen.
FUTURA pretendeert niet overal een oplossing voor te weten, maar denkt dat je door samenwerking sneller je doel bereikt. Wil je iets meer weten? Bel dan (na 18.00 uur) naar 075-314220 of 02982-6547.
Als je de beschikking over een modem hebt kan je inloggen op het FUTURA-BBS, 075-352035 (baudrates: 300, 1200 en 2400).

Stichting Computer Hobyisten Nederland

De eerstvolgende bijeenkomsten van de SCN zijn op 9 april, 14 mee en 4 juni. Inl: 020-934699 (maandag, dinsdag, donderdag tussen 19.30 en 22.00 uur)

VCGN

De VCGN is van plan op 30 april een vlooienmarkt te organiseren. Er worden mensen gevraagd voor de organisatie van een 'Commodore Braderie' op koninginnedag. Iedereen die iets heeft voor de vlooienmarkt wordt verzocht dat aan het secretariaat te melden. Tel: 070-950779.

Kortrijkse Commmodore club

Deze club in het Belgische Kortrijk bestaat inmiddels twee jaar en heeft ongeveer honderd leden, die elke drie weken op zaterdag van 9 tot 12 uur bij elkaar komen in het Groeningeheem, aan de Passionistenlaan. Lidegeld per persoon bedraagt 500 BFr, voor een heel gezin 750 BFr. Daarin zijn inbegrepen gratis lessen en een clubblad. Om alleen het blad te ontvangen volstaat het betalen van een vergoeding voor de verzendkosten. Inl: na 18.00 uur, tel. 056-213981 of schriftelijk aan Danny Heytens, Schilderstraat 34, 8500 Kortrijk.

ture aardig aan zijn/haar trekken. De graphics en het geluid zijn van goede kwaliteit. Stifflip gaat in de winkel rond de *f* 30,- kosten.

Star Paws

Na het succes van Roadrunner lag het in de lijn der verwachtingen dat het genre maffe achtervolgings-games snel zou toenemen. **Star Paws** is daarvan een der betere voorbeelden. De situatie is natuurlijk weer krankzinnig. Op een verlaten planeet leeft de interstellaire munteenheid, de Griffin. Deze beestjes zijn niet alleen waardevol en smakkelijk maar ook nog eens watervlug. De enige manier om hen in de kladden te vatten is met een soort van rugby-tackle en dat vergt veel oefening en snelle reflexen.
Het bestaan van de Griffins wordt bedreigd door een stel onverlaten die de valutahandel wil ontregelen door deze levende munteenheid te vangen en snel door te kweken. Captain Rover Pawstrong moet deze monetaire ramp voorkomen door de Griffins tijdig te vangen. Aanvankelijk is Pawstrong nog een klungel en kost het de grootste moeite om de acht mijnschachten op de Griffin-planeet op hun bewoners te doorzoeken. Het is zaak om voldoende lampen en energie (in de vorm van gebraden kalkoenen) te vergaren anders ligt de held met al die achtervolgingen zo op apegapen. Van overheidswege worden er supplies gedumpd maar die moet men wel zelf zoeken. Er zijn vier subgames met elk een tijdslimiet.
Wie Roadrunner leuk vond zal zich best met Star Paws kunnen vermaken. De prijs voor de C-128 / C-64-versie bedraagt circa *f* 24,-.

Thrust

Voor de C-16 helden brengt Firebird nu het van de C-64 bekende game **Thrust** uit. Als lid van het verzet dient de speler uit de arsenalen van het Intergalactic Empire een aantal pods te stelen. Deze pods zijn noodzakelijk als energiebron voor hun slagslepen. Natuurlijk laat het imperium deze pods niet zonder slag of stoot door het verzet meenemen. Elke planeet waar pods zijn opgeslagen is een ware vesting die door limpet guns verdedigd wordt. Gelukkig beging het imperium de stommiteit om al het ge-schut op slechts een kerncentrale aan te sluiten zodat het vernietiging van deze centrle de vuurkracht van de verdedigers tot nul reduceert.
Wie denkt dat er op los schieten het gewenste resultaat zal hebben heeft het goed mis. Het is de kunst om de kerncentrale tijdelijk buiten gevecht te stellen totdat het pod geroofd is. Daarna mag de centrale gerust ontploffen en dat levert ook nog een bonus op. Voortijdig ontploffen levert daarentegen helemaal niets op. Dus strategisch schieten.
Het buitmaken van pods gebeurt door er overheen te zweven en hen met een tractorstraal naar binnen te trekken. Dezelfde tractorstraal zorgt tevens voor het tanken van de broodnodige brandstof. Succes hangt van de strategie en behendigheid af.
Het spel weet lange tijd te boeien. Helaas zijn geluid en de graphics niet zo best. De lage prijs van minder dan een tientje maakt echter veel goed.

Game Over

De titel **Game over** doet een gamekiller vermoeden, maar het gaat echt om een arcade-game voor de C-64. In een ver gelegen universum ligt de planeet Hypsis waarop ene Arkos in twintig schermen een quest moet vervullen. Bewapend met ploffer en granaten trekt Arkos ten strijde totdat hij een ruimteschip gevonden heeft om naar een volgende wereld af te reizen.
Het spel begint in de gevangenis met robotbewakers. Elke doeltreffend geworpen granaat betekent weer een robotje minder. Het gooien gaat door de vuurknop wat langer ingedrukt te houden. Een korte druk vuurt de blaster af. Deze ploffer is handig voor het uitschakelen van mijnen en, middels het schieten op een soort vaten, extra wapens, energie of een krachtveld te vergaren.
Inplaats van ladders zijn er op een aantal spelschermen liften. Ook schuilen er op verschillende niveaus extra tegenstanders zoals superrobots (moeten 20 maal aange-schoten worden) het reuzenmonster Orko en nog een collectie groene griezels.
Is de planeet Hypsis afgehandeld dan volgt de begroeide planeet Sckunn. Het wapen van keuze is daar een zware laser die, om het moeilijk te maken, slechts over een beperkte energievoorraad beschikt. De vijand bestaat uit energie opslurpende "kangaroes" die tegen de held aanhoppen. Dan zijn er nog wat vervaarlijke vliegmonsters de liesers - friesers. En zo gaat het maar door. De finale bestaat uit een gevecht met een reuzebewaker die men 75 maal moet raken.
Game Over is een arcadegame met fraaie graphics en sterk geluid dat jammer genoeg niet overal even sterk boeit. De diskuitvoering kost *f* 50,-, de cassetteversie *f* 35,-.

Mystery of the Nile

Egypte is een leuk speldecor voor Oosterse avonturen. In **Mystery of the Nile** van Firebird zijn die helden op zoek naar een gestolen antiek juweel. De speler kan elk van de drie helden besturen, maar er is steeds maar een echt actief. De andere twee wandelen rond aan de hand van de C-64.
Aan het begin van het game is de heldin Janet die in een militair complex te Jarga op zoek is naar het juweel. Zij staat op een balkon waaronder de bewakers patrouilleren. Om naar een volgend balkon te springen vergt enig circustalent, maar daar liggen wel een aantal nuttige handgranaten. Met behulp van de handgranaten kan de heldin de soldaten uitschakelen.
Na enkele schermutselingen ontmoet Janet haar Arabische vriend Al-Hasan. Eerst heb je daar bitter weinig aan totdat deze arabier zijn (als zwaard gebruikte) paraplu weet te vinden. Daarna ramt en steekt Al-Hasan er duchtig op los.
Weer later duikt er nog een revolverheld, Nevada, op en kan het soldaatje killen pas echt beginnen. Een van de grootste problemen is de overige twee helden tijdens de eigen doldrieste acties niet om zeep te helpen. Daarzij wat comateus onder de hoede van de C-64 0f C-128 rondwandelen is de kans daarop vrij groot en elk spelfiguur heeft slechts vier levens. Een goede planning voorkomt in deze een voortijdig einde van het spel.
Er zijn meerdere spelniveaus van elk 10 schermen. Aan het einde van een niveau krijgt de speler een wachtwoord voor het volgende niveau.
Qua geluid en graphics is Mystery of the Nile redelijk tot goed verzorgd. Het spel weet best te boeien, maar is soms toch frustrerend als men per vergissing weer een metgezel heeft omgebracht. De cassetteversie kost circa *f* 30,-, de disketteversie *f* 45,-.

In de Programmer's Reference Guide van de C-64 staat de Basic-manier om het beeld punt voor punt te verschuiven, het zogenaamde Smooth Scrollen. Helaas is deze methode behalve langzaam ook niet bepaald trillingsvrij. Rene Janssen bekijkt de werking en geeft een handige tip ter verbetering.

Smooth Scrolling

Voor het echte Smooth Scrollen zul je in machinetaal moeten programmeren, maar ook dan moet je op een paar dingen letten voordat alles goed gaat. Maar als je secuur te werk gaat, en een beetje handigheid hebt opgedaan met het werken in Machinetaal kan iedereen er mee aan het werk.

Smooth Scrollen.

Om het scherm te verschuiven moet je het volgende doen:

° **1)** Verklein het scherm en wel naar 38 kolommen voor horizontaal verschuiven en/of naar 24 rijen voor verticaal verschuiven.

° **2)** Verklein het scrollregister (53270 of 53265) van 7 naar 0 of vergroot het van 0 naar 7 (hangt af van de richting).

° **3)** Verschuif het beeld byte voor byte. Als een regel er bijv. zo uitziet:

*****Rene Janssen*****

en je verschuift het beeld naar links, dan ziet het er zo uit:

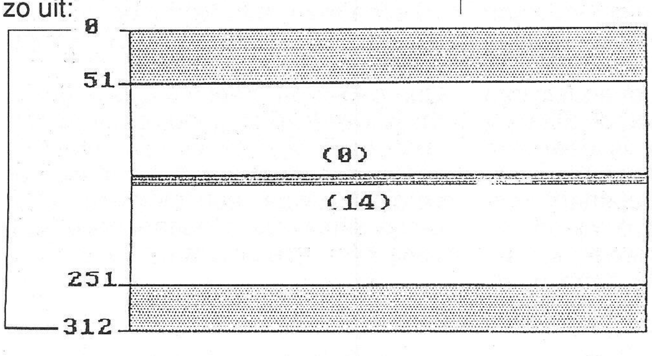
****Rene Jansen*****

4) Stel het scrollregister weer in op zijn beginwaarde en begin opnieuw.

Om dit alles trillingvrij te laten verlopen moeten deze veranderingen gebeuren als het beeldraster buiten beeld is.

Het Beeldraster

Het raster van het beeldscherm is grafich voor te stellen, en ziet er ongeveer zo uit:



Het raster is binnen beeld van 51 tot 251. Veranderingen in het scherm kunnen dus het beste plaatsvinden als het raster zich in gebied 0-51 of 251-312 bevindt. Dit is echter niet zo makkelijk.

Voorbeeld

Om nu te laten zien wat er allemaal mogelijk is, heb ik hier een programma opgenomen, dat het volgende doet: Het verschuift het scherm naar links en vervolgens wordt het beeld geroteerd. Dat betekend dat het beeld zodanig verschoven wordt dat wat er links afgaat er rechts weer bijkomt. Ziet een beeld er bijv. zo uit:

Verschuif het beeld naar links......

Dan ziet het er na een rotatie zo uit:

erschuif het beeld naar links V

Nu neemt het roteren van een regel veel tijd in beslag (ook al lijkt dat niet zo). Na het roteren van 1 regel is het raster nl. al weer 14 regels verder. Zo is het makkelijk om uit te rekenen waar het raster zich bevind als een aantal regels geroteerd is d.m.v. de volgende formule:

R = (N*14) + S

Hierin is N de regel van 0 tot 24, S het startpunt (een rasterregel van 0 tot 312), en R de nieuwe rasterregel.

Stel we beginnen het roteren van het hele beeld als het raster net buiten beeld is, ong. bij 251. Als we bij regel 21 zijn bevindt het raster zich op: (21*14)+251=545.

Als het raster bij 312 is begint het natuurlijk weer bij 0 dus de eigenlijke regel is 545- 312=233. Je ziet dat, als het scherm nog geen eens helemaal geroteerd is, het raster zich toch weer binnen beeld bevind. Als we daar niets aan zouden doen zou het beeld toch nog trillen.

Het programma.

Het eigenlijke programma in machinetaal, staat op de volgende bladzijde (listing), en daarna een Basic-listing van hetzelfde (listing 2).
De werking is als volgt:

$COOO-$C)18:

Hier wordt het scrollregister verkleint van 6 naar 0 waardoor het scherm 6 puntjes naar links verschuift. Dit wordt gedaan steeds als het raster zich op 252 bevindt (register 53266 wordt bekeken).

$CO1A-$CO1F:

Om ons wat tijd te gunnen beginnen we met roteren als het raster zich net onder de eerste regel begind op rasterregel 59.

$CO21-$CO3B:

De roteerroutine.

We beginnen boven (LDX #$00). De Kernalroutine in $E9FO berekent het begin van elke regel. Het regelnummer, van 0 tot 24 moet zich dan in het X-register bevinden. Het beginadres dat dan bekend is (1024 tot 1984) wordt in de pointer $D1/$D2 opgeslagen. Daarna wordt het teken helemaal links opgeslagen in $FB. Dan wordt de hele regel naar links verschoven .

Tenslotte wordt het beginteken op het eind van de regel geplaatst.
Dit is natuurlijk maar een voorbeel van een roteerroutine. Er zijn misschien wel betere routine's, maar dit is in ieder geval wel een van de kleinsten (denk ik).

$C03C-$C03E:

We zijn nu bij regel 19 en het raster bevind zich bij regel :

(19*14)+59 = 325 - 321 = 13

Het raster is weer boven en bijna binnen beeld in 51, dus het wordt tijd dat we nu al het scrollregister resetten op 7 anders gaat het beeld aan de bovenkant 'springen' oftewel trillen. Dit kunnen we doen omdat we toch al bijna beneden bij regel 24 zijn en dat haalt het raster toch niet meer in.

$CO43-$CO47:

We zijn nu bij regel 24 en het raster bevindt zich op:

(24*14)+59 = 395 - 312 = 83

Dat is binnen beeld en je begrijpt dat het belangrijk was dat we van te voren al, bij regel 19, het scrollregister gereset hebben naar 7, want als we dat nu pas gedaan zouden hebben zou het beeld toch nog gaan trillen.

$CO4A-$CO55:

Het scrollregister wordt gereset bij regel 19.

De werking

Hoe ziet dat alles er nu uit? Type het hele scherm maar eens vol en run dan het volgende Basic-programmaatje:

10 Poke 53270, PEEK(53270) AND 247:REM VERKLEINEN SCHERM
20 SYS 49152

Het resultaat: het scherm roteert vrijwel trillingsloos.

We schuiven verder....

Nu is roteren niet de meest toegepaste vorm van Smooth Scrollen. Meestal wordt er in spelletjes gebruik gemaakt van het verschuiven van enorme schermen. Het principe blijft echter hetzelfde. Je moet er alleen maar voor zorgen dat je het scrollregister op tijd reset om trillingen te voorkomen. Ook komt het voor dat er alleen maar gedeelten van het scherm (bijv. de bovenste helft) gescrolld wordt. Hier zul je gebruik moeten maken van zogenaamde Raster- interrupts. Maar daarover misschien een volgende keer.

Rene Janssen

Listing 1. Smooth Scrolling

```
   PC  SR AC XR YR SP
.;803E 32 00 83 00 F6

.,C000  A9 06     LDA #$06
.,C002  85 FB     STA $FB
.,C004  A9 FC     LDA #$FC
.,C006  CD 12 D0  CMP $D012
.,C009  D0 FB     BNE $CA06
.,C00B  AD 16 D0  LDA $D016
.,C00E  29 F8     AND #$F8
.,C010  18        CLC
.,C011  65 FB     ADC $FB
.,C013  8D 16 D0  STA $D016
.,C016  C6 FB     DEC $FB
.,C018  10 EA     BPL $CA04
.,C01A  A9 3B     LDA #$3B
.,C01C  CD 12 D0  CMP $D012
.,C01F  D0 FB     BNE $C01C
```

SCROLL REGISTER VAN 6 NAAR 0.

```
.,C021  A2 00     LDX #$00
.,C023  20 F0 E9  JSR $E9F0
.,C026  A0 00     LDY #$00
.,C028  B1 D1     LDA ($D1),Y
.,C02A  85 FB     STA $FB
.,C02C  C8        INY
.,C02D  B1 D1     LDA ($D1),Y
.,C02F  88        DEY
.,C030  91 D1     STA ($D1),Y
.,C032  C8        INY
.,C033  C0 27     CPY #$27
.,C035  D0 F5     BNE $C02C
.,C037  A5 FB     LDA $FB
.,C039  91 D1     STA ($D1),Y
.,C03B  E8        INX
.,C03C  E0 13     CPX #$13
.,C03E  D0 03     BNE $C043
.,C040  20 4A C0  JSR $C04A
.,C043  E0 19     CPX #$19
.,C045  D0 DC     BNE $C023
.,C047  4C 00 C0  JMP $C000
.,C04A  AD 16 D0  LDA $D016
.,C04D  29 F8     AND #$F8
.,C04F  18        CLC
.,C050  69 07     ADC #$07
.,C052  8D 16 D0  STA $D016
.,C055  60        RTS
```

ROTEREN

SCROLLREGISTER RESETTEN

Listing 2. Basic Smooth Scrolling

```
2000 FORL=0TO5:CX=0:FORD=0TO15:READA:CX=CX+A:POKE49152+L*16+D,A:NEXTD
2010 READA:IFA<>CXTHENPRINT"ERROR IN LINE";2040+(L*10):STOP
2020 NEXTL:END
2040 DATA169,6,133,251,169,252,205,18,208,208,251,173,22,208,41,248,2562
2050 DATA24,101,251,141,22,208,198,251,16,234,169,59,205,18,208,208,2313
2060 DATA251,162,0,32,240,233,160,0,177,209,133,251,200,177,209,136,2570
2070 DATA145,209,200,192,39,208,245,165,251,145,209,232,224,19,208,3,2694
2080 DATA32,74,192,224,25,208,220,76,0,192,173,22,208,41,248,24,1959
2090 DATA105,7,141,22,208,96,0,0,0,0,0,0,0,0,0,0,579

READY.
```

Eind vorig jaar hebben we besloten een prijsvraag uit te schrijven. Dat hebben we geweten, stapels, stapels post. Aan ons is nu de taak deze allemaal te gaan bekijken en uit te proberen. Wat opvalt bij het bekijken van de brieven is het grote aantal inzendingen van onze zuider buren. We hebben nog wel wat tijd nodig voor we de prijswinnaars bekend kunnen gaan maken, maar de prijzen worden in iedergeval op de komende INFO-beurs, 23 april uitgereikt.

R. Goudriaan.

Syntax Checksum C64

Het overtikken van een listing kan een heel karwei zijn en als u een beetje normaal mens bent dan maakt u daarin beslist een aantal fouten. Nu is niets moeilijker om de fouten uit je eigen werk te halen. Al geruime tijd heeft Jan Bodzinga hiervoor een zgn. Checksum-programma geschreven. Om de vele nieuwe lezers van Commodore-info te helpen volgt hieronder nog een keer een volledige uitleg over de werking van dit programma, waarmee het, hoe vreemd dat misschien ook lijkt, echt mogelijk is om met behulp van dit programma de fouten in elke door ons geplaatste listing op te sporen.

Hiervoor gaat u als volgt te werk:

1. U tikt de listing heel zorgvuldig over en SAVEt hem voordat u het programma RUNt op een diskette of cassette.
2. U tikt het RUN commando in. Mocht het programma de boodschap 'FOUT in dataregels!' geven dan heeft u een fout bij het overtikken gemaakt. Herstel dan de fout en SAVE de verbeterde versie. Mocht het programma met de boodschap 'data is weggezet checksum testen met sys...'komen dan is tot dusver alles goed. Het programma is nu in een stukje machinetaalgeheugen gezet. Als u het NEW commando geeft blijft het toch in de computer staan.

Alle door ons geplaatste programma's zijn in Basic geschreven.

Als u een programma heeft overgetikt SAVE het eerst, mocht er iets mis gaan dan hoeft u niet de gehele listing opnieuw te gaan intikken. Als u nu een programma op fouten wilt gaan controleren dan kunt u dat in het geheugen laden(wel eerst het checksum programma hebben gerund). Vervolgens typt u zonder het programma te runnen de opdracht sys 49152(c-64).

Als alles goed is gegaan loopt er nu een rij regelnummers over het scherm met getallen erachter. Dezelfde lijst staat ook achter elk door ons geplaatste programma. Wijkt nu een nummer achter een regelnummer af van het nummer dat in het blad staat dan heeft u in die regel iets anders ingetikt dan er in het blad stond. U kunt de stroom getallen d.m.v. de RUN/STOP toets pauzeren en weer vervolgen met de F1 of F7 toets. Het is uitermate belangrijk dat u goed met dit programma overweg kunt en mocht u het niet goed werkend krijgen bel dan gerust even met onze listingservice telefoonlijn. (Maandag 17.00 - 21.00 uur. Telefoonnummer 02155-25162.)

```
1    rem ******************************
     ***
2    rem basic loader "SYNTAX.CHECKSUM"
3    rem na de commando's "run" en "new
     "
4    rem blijft dit programma in het ge
     -
5    rem heugen. laad het te testen pro
     -
6    rem gramma en tik daarna sys 49152
     .
7    rem ******************************
     ***
10   i=49152  :rem beginadres
20   reada:ifa<0then40:rem data ingelez
     en
30   pokei,a:i=i+1:b=b+a:goto20
40   if b<>16844thenprint"[SHIFT-CLR]fo
     ut[SPACE]in[SPACE]dataregels!":b=0
     :end
50   poke49184,148:poke49185,192
55  i=49300
60   read a: ifa<0then80
70   pokei,a:b=a+b:i=i+1:goto60
80   if b<>20068thenprint"[SHIFT-CLR]fo
     ut[SPACE]in[SPACE]dataregels![SPAC
     E](vanaf[SPACE]regel[SPACE]240)":b
     =0:end
90   print"data[SPACE]is[SPACE]weggezet
     "
95  print"checksum[SPACE]testen[SPACE]
     met[SPACE]sys49152"
100  data 165,43,166,44,133,163,134,164
     ,169,147
110  data 32,210,255,160,0,240,3,32,73,
     192
120  data 32,73,192,208,1,96,32,225,255
     ,208
130  data 3,76,116,164,32,81,192,32,73,
     192
140  data 240,12,201,32,240,247,24,101,
     167,133
150  data 167,76,37,192,166,167,169,0,1
     32,168
160  data 32,205,189,169,13,32,210,255,
     164,168
170  data 76,17,192,200,208,2,230,164,1
     77,163
180  data 96,162,0,189,123,192,240,6,32
     ,210
190  data 255,232,208,245,32,73,192,170
     ,32,73
200  data 192,132,168,32,205,189,162,3,
     169,32
210  data 32,210,255,202,208,250,169,0,
     133,167
220  data 164,168,96,82,69,71,69,76,32,
     0
230  data  -1
240  data 165,197,201,3,240,7,201,4,240
250  data 6,76,148,192,76,34,192,169
260  data 147,32,210,255,76,161,192
270  data -1
```

** EINDE LISTING checksum 64 **

REGEL	1	249	REGEL	90	245
REGEL	2	84	REGEL	95	237
REGEL	3	105	REGEL	100	183
REGEL	4	2	REGEL	110	158
REGEL	5	246	REGEL	120	232
REGEL	6	152	REGEL	130	183
REGEL	7	249	REGEL	140	96
REGEL	10	157	REGEL	150	96
REGEL	20	64	REGEL	160	127
REGEL	30	38	REGEL	170	71
REGEL	40	57	REGEL	180	223
REGEL	50	14	REGEL	190	73
REGEL	55	251	REGEL	200	79
REGEL	60	192	REGEL	210	109
REGEL	70	42	REGEL	220	106
REGEL	80	244	REGEL	230	225

Memorytus 64

Het spel dat u verkrijgt als u de volgende listing intypt behoeft eigenlijk geen uitleg. Een geheugenspel waarbij de kinderen vaak slimmer zijn dan de ouders.

```
10    c$="[20xCRSR-DOWN]"
20    d=int(rnd(0)*9+1)
30    poke 53280,3:poke 53281,3
40    print"[SHIFT-CLR][4xCRSR-DOWN]":fo
      r a=1 to 3:print spc(4) "memoritus
      ";:next a:print
50    for b=1 to 9:hl$=hl$+chr$(113):nex
      t b
60    for c=1 to 3:print spc(4) hl$;:nex
      t c:print"[5xCRSR-DOWN]"
70    print spc(13) "a.[SPACE]memory[SPA
      CE]twee":print
80    print spc(13) "b.[SPACE]memory[SPA
      CE]drie":print"[3xCRSR-DOWN]"
90    input"[4xSPACE]welke[SPACE](a/b)";
      i$
100   if i$="a" then c=1:goto 130
110   if i$="b" then c=2:goto 140
120   if i$<>"a" or i$<>"b" then 90
130   a$="qwertyasdfghzxcvbn":b$="839715
      425974861362":goto 150
140   a$="qwertyuiopasdfghjklzxcvbnm,":b
      $="347261321596475987248851639"
150   poke 53280,0:poke 53281,0:print"[S
      HIFT CLR]"
160   dim d$(30),e$(30),f$(3),aa(3)
170   on c gosub 220,470
180   on c gosub 270,520
190   on c gosub 320,570:on c gosub 440,
      690:if tl<9 then 180
200   goto 730
210   end
220   for n=1 to 18:d$(n)=mid$(a$,n,1)
230   nr=n+d:if nr<19 then 250
240   nr=nr-18
250   e$(n)=mid$(b$,nr,1)
260   next n:return
270   for k=1 to 3:for j=1 to 6
280   print"[HOME]"
290   print right$(c$,5*k);tab(5*j);
300   print d$((k-1)*6+j)
310   next j,k:return
320   print:f$(2)=""
330   for i=1 to 2
340   get ky$
350   if ky$="" or ky$=f$(1) then 340
360   f$(i)=ky$:ke=ke+1
370   aa(i)=0:for n=1 to 18
380   e=int((n-.5)/6)+1:f=n-(e-1)*6
390   if d$(n)<>f$(i) then 420
400   print"[HOME]";right$(c$,5*e+1);tab
      (5*f);
410   print e$(n);:aa(i)=n
420   next n:if aa(i)=0then 340
430   next i:return
440   if e$(aa(1))<>e$(aa(2)) then retur
      n
450   d$(aa(1))="[SPACE]":d$(aa(2))="[SP
      ACE]"
460   tl=tl+1:return
470   for n=1 to 27:d$(n)=mid$(a$,n,1)
480   nr=n+d:if nr<28 then 500
490   nr=nr-27
500   e$(n)=mid$(b$,nr,1)
510   next n:return
520   for k=1 to 3:for j=1 to 9
530   print"[HOME]"
540   print right$(c$,5*k);tab(4*j);
550   print d$((k-1)*9+j)
560   next j,k:return
570   print:f$(2)="":f$(3)=""
580   for i=1 to 3
590   get ky$
600   if ky$="" or ky$=f$(1) then 590
610   f$(i)=ky$:ke=ke+1
620   aa(i)=0:for n=1 to 27
630   e=int((n-.5)/9)+1:f=n-(e-1)*9
640   if d$(n)<>f$(i) then 670
650   print"[HOME]";right$(c$,5*e+1);tab
      (4*f);
660   print e$(n);:aa(i)=n
670   next n:if aa(i)=0then 590
680   next i:return
690   if e$(aa(1))<>e$(aa(2)) then retur
      n
700   if e$(aa(2))<>e$(aa(3)) then retur
      n
710   d$(aa(1))="[SPACE]":d$(aa(2))="[SP
      ACE]":d$(aa(3))="[SPACE]"
720   tl=tl+1:return
730   print"[SHIFT-CLR][6xCRSR-DOWN]" sp
      c(10) "u[SPACE]heeft[SPACE]er";ke;
      "beurten"
740   print:print spc(15) "overgedaan":p
      rint"[4xCRSR-DOWN]"
750   input "nog[SPACE]een[SPACE]keertje
      [SPACE](j/n)[SPACE]";m$
760   if m$="j" then run
770   if m$="n" then 210
780   if m$<>"j" or m$<>"n" then 750
```

** EINDE LISTING memorytu **

REGEL	10	177	REGEL	200	35
REGEL	20	248	REGEL	210	128
REGEL	30	43	REGEL	220	9
REGEL	40	84	REGEL	230	78
REGEL	50	52	REGEL	240	6
REGEL	60	178	REGEL	250	100
REGEL	70	96	REGEL	260	152
REGEL	80	199	REGEL	270	72
REGEL	90	236	REGEL	280	240
REGEL	100	83	REGEL	290	223
REGEL	110	86	REGEL	300	160
REGEL	120	248	REGEL	310	11
REGEL	130	120	REGEL	320	182
REGEL	140	147	REGEL	330	131
REGEL	150	207	REGEL	340	105
REGEL	160	55	REGEL	350	157
REGEL	170	188	REGEL	360	101
REGEL	180	189	REGEL	370	247
REGEL	190	77	REGEL	380	231

REGEL 390	55	REGEL 590	105
REGEL 400	66	REGEL 600	164
REGEL 410	50	REGEL 610	101
REGEL 420	209	REGEL 620	247
REGEL 430	147	REGEL 630	237
REGEL 440	161	REGEL 640	62
REGEL 450	161	REGEL 650	65
REGEL 460	149	REGEL 660	50
REGEL 470	9	REGEL 670	216
REGEL 480	76	REGEL 680	147
REGEL 490	6	REGEL 690	161
REGEL 500	100	REGEL 700	163
REGEL 510	152	REGEL 710	144
REGEL 520	75	REGEL 720	149
REGEL 530	240	REGEL 730	1
REGEL 540	222	REGEL 740	28
REGEL 550	163	REGEL 750	83
REGEL 560	11	REGEL 760	109
REGEL 570	212	REGEL 770	122
REGEL 580	132	REGEL 780	70

Windowla 64

Het volgende programma is er één van Ronald Sloot. Ook al geen onbekende in de programmeer wereld. Om dit programma te begrijpen en te gebruiken in eigen programma's moet u wel een redelijke kennis van het programmeren te hebben. Het gehele programma-bestaat uit twee programma's die ook los zijn te gebruiken. Windowla en Sprite extension. De startadressen zijn: Sprite-extension $ccec-ce7b en Windowla $ce7c-cfff. Ze zijn opgenomen in een basiclisting voor de commodore 64 in data-hex vorm. In het programma is een demo opgenomen die alle mogelijkheden laat zien. Een eventuele powercardridge moet worden uitgeschakeld, dit gebeurt ook in de basiclisting. Na het runnen verschijnt er een handleiding op het scherm.

```
10     rem
20     rem    window progamma
30     rem     by r.sloot
40     rem      1987 pce
50     rem  voor commodore info
60     rem
70     rem
80     ifpeek(777)=222then:rename:rem pow
       er
90     de=8:s=52860:ba=52460:poke53280,6:
       poke53281,6:gosub380:fori=0to23:re
       adg$
100    forb=1tolen(g$)step2:h$=mid$(g$,b,
       2):i$=left$(h$,1):gosub1090
110    c=c+16*e:i$=right$(h$,1):gosub1090
       :c=c+e:pokeba+d,c:d=d+1:t=t+c:c=0:
       nextb,i
120    ift<>110958ord<>787thenprint"[CRSR
       -DOWN]missing[SPACE]data:"787-d"[S
       PACE]tegen"110958-t:stop
130    fori=0to23:reada:poke704+i,a:next:
       fori=0to39:poke728+i,0:next:rem sp
       rite 0
140    rem
150    rem demo
160    rem
170    sys53139,1:rem scherm opslag
180    syss,4,3,3,1,23,14,a$
190    gosub1050:ifk$<"a"ork$>"f"then190
200    a=asc(k$)+128:t$=chr$(a)
210    syss,11,10,14,15,25,6,b$+t$+"[pijl
       links][4xSPACE]Is[SPACE]this[SPACE
       ]Alright[pijl links][8xSPACE](y/n)
       ?"
220    gosub1050:ifk$="n"then180
230    syss,8,15,7,1,5,4,"[pijl links][SP
       ACE]Ok!":gosub1110
240    syss,6,2,15,14,3,10,"[2x pijl link
       [SPACE]E[pijl links][SPACE]i[pijl
       links][SPACE]n[pijl links][SPACE]d
       [pijl links][SPACE]e":gosub1110:sy
       s53139,0
250    syss,3,2,14,1,30,5,m1$:gosub1110:g
       osub1110:syss,5,4,1,7,16,18,m2$
260    sysba,0,150,163,111,320,190,6,22:s
       ys53139,0:gosub1070
270    syss,3,3,3,1,32,9,de$:sysba,14,100
       ,160,120,220,136,7,27
280    ifpeek(2)=1thensys53139,0:end
290    gosub1070:syss,14,9,14,14,6,5,"[pi
       jl links][SPACE]Tape[2xSPACE]Disk"
       :sysba,1,80,185,135,206,151,15,21
300    poke53269,0:ifpeek(2)=0thende=1
310    poke2,de:print"[SHIFT-CLR]"tab(164
       )"Druk[SPACE]spatie[SPACE]wanneer[
       SPACE]u[SPACE]klaar[SPACE]bent"
320    printtab(4)"Herladen[SPACE]-"chr$(
       13)"[2xSPACE]10[SPACE]x=x+1:ifx=1t
       henload'naam',device,1"
330    wait197,32:rem wacht op space
340    poke43,236:poke44,204:poke45,255:p
       oke46,207:save"windowla[SPACE]v1.0
       ",peek(2),1:end
350    rem
360    rem commando uitleg
370    rem
380    print"[SHIFT-CLR][2xCRSR-DOWN]"chr
       $(14)tab(7)"Commando[SPACE]Uitleg[
       SPACE]Windowla"
390    print"[2xCRSR-DOWN]sys52860,x-posi
       tie,y-positie,randkleur":printtab(
       8)",binnenkleur";
400    print",breedte,lengte":printtab(8)
       ",string[SPACE]of[SPACE]"chr$(34)"
       tekst"chr$(34)
410    print"[CRSR-DOWN]SYS53139,0[SPACE]
       =[SPACE]haal[SPACE]op[SPACE](scher
       m+kleuren)":printtab(8)",1[SPACE]=
       [SPACE]sla[SPACE]op";
420    print"[SPACE](scherm+kleuren)":pri
       nt"[CRSR-DOWN]SYS52460,spritekleur
       ,snelheid,x1,y1"
430    printtab(8)",x2,y2[SPACE]en[SPACE]
       dan[SPACE]KUNT[SPACE]u[SPACE]er[SP
       ACE]ook":printtab(8)"de[SPACE]gege
       vens[SPACE]voor[SPACE]de"
440    printtab(8)"meelopende[SPACE]balk[
       SPACE]nog[SPACE]bijzetten":printta
       b(8)"Nml:,beginpositie";
450    print",eindpositie":print"[CRSR-DO
       WN]x1,y1,x2,y2[SPACE]geven[SPACE]h
       et[SPACE]bewegingsvlak[SPACE]aan"
460    printtab(8)"Joystick[SPACE]besturi
       ng[SPACE]p2":print"[CRSR-DOWN]"tab
       (7)"Demo[SPACE]in[SPACE]plm.[SPACE
       ]55[SPACE]seconden"
470    rem
480    rem demo strings
490    rem het teken [pijl links] in een
       string
500    rem betekent een return
510    rem
520    a$="[pijl lionks][4xSPACE]Disk[SPA
       CE]Operaties[pijl links]----------
```

```
     -------------[SPACE]a)[SPACE]Forma
     teren [SPACE]b)[SPACE]Load"
530  a$=a$+"[pijl links[SPACE]c)[SPACE]
     Scratchen[pijl links][SPACE]d)[SPA
     CE]Save[pijl links][SPACE]e)[SPACE
     ]Replace[pijl links][SPACE]f)[SPAC
     E]Verify[pijl links
     --------------------"
540  a$=a$+"---[5xSPACE]Press[SPACE]a[S
     PACE]key[pijl links][3xSPACE]Betwe
     en[SPACE]A[SPACE]and[SPACE]F"
550  b$="[pijl links][3xSPACE]You[SPACE
     ]have[SPACE]chosen[SPACE]"
560  m1$="[pijl links][2xSPACE]Hierna[S
     PACE]volgt[SPACE]demo[SPACE]Nummer
     [SPACE]2[pijl links][SPACE]M.b.v[S
     PACE]Sprite[SPACE]extension[SPACE]
     v.1.0[pijl links]"
570  m2$="[pijl links][3xSPACE]Bladen[S
     PACE]c64[pijl links]--------------
     --[SPACE]Commdore[SPACE]Info[pijl
     links][SPACE]Dossier[pijl links][S
     PACE]64'er[pijl links][SPACE]Zzap'
     64"
580  m2$=m2$+"[pijl links][SPACE]Comput
     e's[pijllinks][SPACE]Run[pijl link
     s] [SPACE]Your[SPACE]Commodore[pij
     l links][SPACE]Ram[pijl links][SPA
     CE]Mcn[SPACE]Magazine[pijl links][
     SPACE]Commodore"
590  m2$=m2$+"[pijl links]-------------
     ---[SPACE]Maakt[SPACE]u[SPACE]A.U.
     B[pijl links][2xSPACE]uw[SPACE]keu
     ze..?"
600  de$="[pijl links[SPACE]En[SPACE]nu
     [SPACE]gaan[SPACE]we[SPACE]dit[SPA
     CE]programma[pijl links][SPACE]sav
     en.[SPACE]U[SPACE]verliest[SPACE]D
     AN[SPACE]uw[SPACE]basic[pijl links
     ][SPACE]listing."
610  de$=de$+"[2x pijl links][6xSPACE]A
     kkoord[pijl links][6xSPACE]Niet[SP
     ACE]Akkoord":return
620  rem
630  rem sprite expansion voor
640  rem bevoorbeeld dit programma
650  rem dit kan ook los worden
660  rem gebruikt ($ccec-$ce7b)
670  rem
680  dataa5d385a8a5d685a9a90b8df807ad15
     d009018d15d020decd8e27d020decd8693
     205bce
690  data86fd85bb84bc8e01d08d00d08c10d0
     205bce86fe85fb84fca000a92cd17af007
     a90085
700  data944c46cda901859485b920decd869d
     20decd86b7a209ad00dca4932400240024
     002400
710  datac8ead0f4dd64cdf00fca10f84c46cd
     777b7e7d7a7675796f20c0cda594d0034c
     46cd38
720  dataad01d0e9328595184ab0204ab01d4a
     b01a8597a5b9d02aa59cc597f00a20a6cd
     a59785
730  data9c20a6cda90085b94c46cd85d6206c
     e5a49db1d1498091d1c8c4b7d0f560a597
     85024c
740  data98cd8a0aaabdcdcd48bdcccd4860ec
     cd0ccee4cde8cd41ce2cce33ce3ace48ce
     20fdae
750  data209eb7602066ce602071ce60ad00d0
     c5fbd008ad10d0c5fcd001601869018d00
     d09008
760  dataad10d049018d10d060ad00d0c5bbd0
     08ad10d0c5bcd0016038e9018d00d0b008
     ad10d0
770  data49018d10d06020edcd2066ce6020ed
     cd2071ce60200dce2071ce60200dce2066
     ce60a6
780  dataa9a4a81820f0ff686838a597e50285
     026020fdae20ebb7a514a41560ad01d0c5
     fdf003
790  datace01d060ad01d0c5fef003ee01d060
800  rem
810  rem window data
820  rem kan ook los gebruikt worden
830  rem plaats ($ce7c-$cfff)
840  rem
850  dataa5d385a8a5d685a9ad860285aa2062
     cf86b02062
860  datacf86b12062cf86be2062cf86b42062
     cf86b52062cf86b620fdae209ead20a3b6
     85b7a5
870  data22a62385fb86fc2077cfa970a4d320
     70cfc8a200a9402070cfc8e8e4b5d0f5a9
     6e2070
880  datacfa90085b8a90d20d2ffa4b084d3a9
     5d2070cfa200a9a0c82069cfe8e4b5d0f5
     c8a95d
890  data2070cfe6b8a5b8c5b6d0d7a96da4b0
     2070cfc8a200a9402070cfc8e8e4b5d0f5
     a97d20
900  data70cfe6b0e6b12077cfa5be8d8602a0
     00a200b1fbc95ff02048a91220d2ff6820
     d2ffc8
910  datac4b7f03de8e4b5d0e6a90d20d2ffa5
     b085d34c2ccfa90d20d2ffa5b085d3a2ff
     4c3ecf
920  data20fdae209eb76091d1a5be91f36091
     d1a5b491f360a4b0a6b11820f0ff60a992
     20d2ff
930  dataa6a9a4a81820f0ffa5aa8d86026020
     62cfe000f005e001f01260a9b485bca9d8
     85bea9
940  datab3a2b0a0044cbecfa9d885bca9b485
     bea907a204a0b085baa90085fb85fd85bb
     85bda9
950  data36850186fc84fea000b1fb91fdb1bb
     91bdc8d0f5e6fce6fee6bce6bea5fcc5ba
     d0e7b1
960  datafb91fdb1bb91bdc8c0e8d0f3a93785
     0160
970  rem
980  rem sprite data
990  rem opgeslagen in blok 11(*64)
1000 rem
1010 data254,,,252,,,248,,,252,,,254,,,
     207,,,135,128,,3,,
1020 rem
1030 rem losse routines
1040 rem
1050 getk$:ifk$=""then1050
1060 return
1070 ifpeek(56320)=111then1070
1080 return
1090 e=val(i$):ifasc(i$)<58thenreturn
1100 e=(asc(i$)-55):return
1110 forwa=0to1000:next:return
```

** EINDE LISTING windowla **

REGEL	10	143
REGEL	20	187
REGEL	30	59
REGEL	40	64
REGEL	50	166
REGEL	60	143
REGEL	70	143
REGEL	80	154
REGEL	90	15
REGEL	100	26
REGEL	110	187
REGEL	120	201
REGEL	130	130
REGEL	140	143
REGEL	150	180
REGEL	160	143
REGEL	170	81
REGEL	180	31
REGEL	190	90
REGEL	200	123
REGEL	210	201
REGEL	220	11
REGEL	230	252
REGEL	240	18
REGEL	250	17
REGEL	260	5
REGEL	270	227
REGEL	280	19
REGEL	290	2
REGEL	300	251
REGEL	310	72
REGEL	320	185
REGEL	330	15
REGEL	340	166
REGEL	350	143
REGEL	360	167
REGEL	370	143
REGEL	380	83
REGEL	390	51
REGEL	400	111
REGEL	410	131
REGEL	420	66
REGEL	430	146
REGEL	440	17
REGEL	450	9
REGEL	460	129
REGEL	470	143
REGEL	480	222
REGEL	490	140
REGEL	500	153
REGEL	510	143
REGEL	520	66
REGEL	530	26
REGEL	540	107
REGEL	550	28
REGEL	560	11
REGEL	570	176
REGEL	580	79
REGEL	590	54
REGEL	600	71
REGEL	610	165
REGEL	620	143
REGEL	630	97
REGEL	640	63
REGEL	650	240
REGEL	660	193
REGEL	670	143
REGEL	680	114
REGEL	690	100
REGEL	700	236
REGEL	710	166
REGEL	720	43
REGEL	730	63
REGEL	740	148
REGEL	750	219
REGEL	760	243
REGEL	770	7
REGEL	780	27
REGEL	790	65
REGEL	800	143
REGEL	810	129
REGEL	820	108
REGEL	830	49
REGEL	840	143
REGEL	850	110
REGEL	860	94
REGEL	870	58
REGEL	880	139
REGEL	890	114
REGEL	900	88
REGEL	910	228
REGEL	920	58
REGEL	930	55
REGEL	940	180
REGEL	950	34
REGEL	960	86
REGEL	970	143
REGEL	980	128
REGEL	990	122
REGEL	1000	143
REGEL	1010	125
REGEL	1020	143
REGEL	1030	142
REGEL	1040	143
REGEL	1050	167
REGEL	1060	142
REGEL	1070	82
REGEL	1080	142
REGEL	1090	24
REGEL	1100	169
REGEL	1110	228

Renummer 64

Peter Verplaetse is een van onze vaste Belgische inzenders dit maal met een handige renumber routine. Het is geen gewone routine maar eentje die automatisch alle sprongintructies verandert. Het programma intikken, wegschrijven en runnen. Om de routine te activeren is het intikken van sys 49152 voldoende. Hierna typt u achtereenvolgens in, gescheiden door een komma, de startregel en de stapgroote.

```
10    poke53280,12:poke53281,12:poke5327
      2,23
20    print"[SHIFT-CLR]"tab(16)"[CTRL-7]
      RENUMBER"
30    print"[CRSR-DOWN][CTRL-2]Deze[SPAC
      E]routine[SPACE]hernummert[SPACE]d
      e[SPACE]regelnummers"
40    print"en[SPACE]verandert[SPACE]all
      e[SPACE]spronginstrukties."
50    print"[CTRL-8][CRSR-DOWN][SPACE]-s
      tarten[SPACE]met[SPACE]<SYS49152,s
      r,sg>"
60    print"[4xSPACE]-sr[SPACE]=[SPACE]d
      e[SPACE]eerste[SPACE]regel[SPACE]v
      /h[SPACE]nieuwe"
70    printtab(10)"programma[2xSPACE](0-
      65535)"
80    print"[4xSPACE]-sg[SPACE]=[SPACE]h
      et[SPACE]verschil[SPACE]tussen[SPA
      CE]elke"
90    printtab(10)"nieuwe[SPACE]regel[SP
      ACE](0-255)"
100   print"[COM-2][CRSR-DOWN][SPACE]-fo
      utmeldingen[SPACE]:"
110   print"[4xSPACE]-overflow[SPACE]:[S
      PACE]als[SPACE]regelnummer[SPACE]6
      3999"
120   printtab(16)"overschreden[SPACE]wo
      rdt."
130   print"[4xSPACE]-andere[SPACE]error
      s[SPACE]:[SPACE]fout[SPACE]spronga
      dres."
140   print"[4xSPACE]-na[SPACE]een[SPACE
      ]foutmelding[SPACE]is[SPACE]het[SP
      ACE]progr."
150   print"[5xSPACE]nog[SPACE]niet[SPAC
      E]veranderd."
160   print"[CRSR-DOWN][COM-4][SPACE]-de
      [SPACE]snelheid[SPACE]varieert[SPA
      CE]met[SPACE]een[SPACE]fractie"
170   print"[2xSPACE]v/e[SPACE]seconde[S
      PACE]tot[SPACE]meer[SPACE]dan[SPAC
      E]10[SPACE]minuten."
180   print"[CRSR-DOWN][COM-8][SPACE]-he
      t[SPACE]progr.[SPACE]blijft[SPACE]
      na[SPACE]<NEW>[SPACE]bestaan."
190   printtab(10)"[CRSR-DOWN][CTRL-7]7[
      SPACE]sec.[SPACE]geduld[SPACE]a.u.
      b.[CRSR-UP]"
200   for a=49152 to 49855
210   read q:poke a,q
220   x=x+q:next
230   print"[30xSPACE][2xCRSR-UP]"
240   if x=94698 then printtab(15)"DATA[
      SPACE]O.K.[CRSR-UP]":end
250   printtab(13)"FOUT[SHIFT-SPACE]IN[S
      HIFT SPACE]DATA.[CRSR-UP]":end
260   data120,32,253,174,32,235,183,165,
      20,164,21,141,52,3,133
270   data251,140,53,3,132,252,142,54,3,
      165,122,141,55,3,165
280   data123,141,56,3,169,43,133,253,16
      9,0,133,254,160,0,177
290   data253,170,200,177,253,134,253,13
      3,254,177,253,208,5,136,177
300   data253,240,28,165,252,201,250,144
      ,5,162,15,76,55,164,165
310   data251,24,109,54,3,133,251,165,25
      2,105,0,133,252,76,42
320   data192,169,43,133,253,169,0,133,2
      54,160,0,177,253,170,200
330   data177,253,134,253,133,254,177,25
      3,208,5,136,177,253,240,91
340   data160,2,177,253,133,57,200,177,2
      53,133,58,200,177,253,240
350   data219,201,137,240,24,201,141,240
      ,20,201,167,208,239,200,177
360   data253,201,32,240,249,201,48,144,
      229,201,58,176,225,136,200
```

```
370     data132,2,165,253,24,101,2,133,122
        ,165,254,105,0,133,123
380     data32,138,173,32,247,183,32,19,16
        6,176,5,162,17,76,55
390     data164,165,122,56,229,253,168,177
        ,253,201,44,240,212,76,132
400     data192,169,43,133,251,169,0,133,2
        52,160,0,177,251,170,200
410     data177,251,133,252,134,251,177,25
        1,208,8,136,177,251,208,3
420     data76,67,194,160,3,200,177,251,24
        0,225,201,141,240,24,201
430     data137,240,20,201,167,208,239,200
        ,177,251,201,32,240,249,201
440     data48,144,229,201,58,176,225,136,
        200,132,2,165,251,24,101
450     data2,133,122,165,252,105,0,133,12
        3,32,138,173,32,247,183
460     data169,43,133,253,169,0,133,254,1
        73,52,3,141,57,3,173
470     data53,3,141,58,3,160,0,177,253,17
        0,200,177,253,133,254
480     data134,253,160,2,177,253,197,20,2
        08,7,200,177,253,197,21
490     data240,21,173,57,3,24,109,54,3,14
        1,57,3,173,58,3
500     data105,0,141,58,3,76,64,193,173,5
        8,3,172,57,3,32
510     data149,179,32,142,194,32,221,189,
        165,122,56,229,251,56,229
520     data2,168,140,255,3,165,45,56,237,
        255,3,133,45,165,46
530     data233,0,133,46,165,251,24,101,2,
        133,253,165,252,105,0
540     data133,254,177,253,160,0,145,253,
        172,255,3,165,253,197,45
550     data208,6,165,254,197,46,240,9,230
        ,253,208,231,230,254,76
560     data166,193,160,0,200,185,0,1,208,
        250,136,140,255,3,165
570     data45,133,253,165,46,133,254,160,
        0,177,253,172,255,3,145
580     data253,165,251,24,101,2,141,254,3
        ,165,252,105,0,197,254
590     data208,7,173,254,3,197,253,240,11
        ,165,253,208,2,198,254
600     data198,253,76,216,193,165,45,24,1
        09,255,3,133,45,165,46
610     data105,0,133,46,172,255,3,165,251
        ,24,101,2,133,253,165
620     data252,105,0,133,254,136,185,1,1,
        145,253,152,208,247,32
630     data51,165,165,2,24,109,255,3,133,
        2,164,2,177,251,201
640     data44,208,3,76,22,193,76,246,192,
        173,52,3,141,57,3
650     data173,53,3,141,58,3,169,43,133,2
        51,169,0,133,252,160
660     data0,177,251,170,200,177,251,133,
        252,134,251,177,251,208,5
670     data136,177,251,240,66,160,2,173,5
        7,3,145,251,200,173,58
680     data3,145,251,173,57,3,24,109,54,3
        ,141,57,3,173,58
690     data3,105,0,141,58,3,76,87,194,32,
        43,188,144,26,169
700     data145,141,62,3,169,0,141,63,3,14
        1,64,3,141,65,3
710     data141,66,3,160,3,169,62,32,103,1
        84,96,173,55,3,133
720     data122,173,56,3,133,123,169,255,1
        33,57,133,58,88,96
```

** EINDE LISTING renumber **

REGEL	10	240	REGEL	370	105
REGEL	20	102	REGEL	380	60
REGEL	30	231	REGEL	390	90
REGEL	40	165	REGEL	400	229
REGEL	50	85	REGEL	410	41
REGEL	60	70	REGEL	420	221
REGEL	70	102	REGEL	430	112
REGEL	80	37	REGEL	440	228
REGEL	90	144	REGEL	450	171
REGEL	100	187	REGEL	460	42
REGEL	110	239	REGEL	470	81
REGEL	120	86	REGEL	480	194
REGEL	130	132	REGEL	490	42
REGEL	140	228	REGEL	500	52
REGEL	150	186	REGEL	510	57
REGEL	160	165	REGEL	520	1
REGEL	170	74	REGEL	530	62
REGEL	180	169	REGEL	540	250
REGEL	190	34	REGEL	550	200
REGEL	200	44	REGEL	560	72
REGEL	210	103	REGEL	570	192
REGEL	220	25	REGEL	580	126
REGEL	230	255	REGEL	590	151
REGEL	240	186	REGEL	600	164
REGEL	250	181	REGEL	610	70
REGEL	260	119	REGEL	620	126
REGEL	270	21	REGEL	630	234
REGEL	280	85	REGEL	640	157
REGEL	290	141	REGEL	650	36
REGEL	300	191	REGEL	660	25
REGEL	310	76	REGEL	670	141
REGEL	320	235	REGEL	680	99
REGEL	330	99	REGEL	690	162
REGEL	340	32	REGEL	700	31
REGEL	350	108	REGEL	710	198
REGEL	360	74	REGEL	720	220

Ballon expeditie 64

Door D. Murray uit Termunterzijl werd dit programma ingezonden. Met een joystick in poort twee bestuur je een luchtballon door allerlei opstakels, er zijn verschillende hindernissen die steeds verraderlijker worden.

```
10      rem "[7xDEL][3xSPACE]a[SPACE]chall
        enge[SPACE]game
20      dima$(34)
30      poke53280,0:poke53281,0:fort=54272
        to54296:poket,0:next :ls=10000:ls$
        ="d.w.m"
40      goto1480
50      print"[SHIFT-CLR]":b$="[3xCRSR-DOW
        N]":la=200:sc=0:la(1)=3:rs=0
60      restore:fort=0to34:reada$(t):next:
        poke53281,0:poke53280,0:poke646,14
70      fort=12288+256to12288+256+7:poket,
        0:next
80      fort=0to23:readn:poket+12288,n:nex
        t
90      poke53270,peek(53270)or16:poke5328
        2,2:poke53283,10
100     poke53272,(peek(53272)and240)+12
110     v=53248:pokev+21,1:pokev+0,100:pok
        ev+1,211:poke2040,13:pokev+28,3
120     fort=832to832+62:readn:poket,n:nex
        t
130     pokev+37,6:pokev+38,11:pokev+39,8
140     fort=832+64to832+64+62:readn:poket
        ,n:next :pokev+31,0
150     fort=832+128to832+128+62 :readn:po
        ket,n  :next:
```

```
160   pokev+40,10:poke2041,15
170   data"@[10xSPACE]@[5xSPACE]@[9xSPAC
      E]@"
180   data"@[4xSPACE]@[8xSPACE]@[7xSPACE
      ]@[4xSPACE]@"
190   data"@[17xSPACE]@b[CTRL-8]aaaaaa[C
      OM 7]b@"
200   data"@[26xSPACE]@"
210   data"@b[CTRL-2]aaaaa[COM-7]b@[18xS
      PACE]@"
220   data"@@@[11xSPACE]@@@[10xSPACE]@"
230   data"@[26xSPACE]@"
240   data"@[SPACE]@@[9xSPACE]@@[7xSPACE
      ]@@[3xSPACE]@"
250   data"@[26xSPACE]@"
260   data"@[8xSPACE]@@[6xSPACE]@@[8xSPA
      CE]@"
270   data"@[26xSPACE]@"
280   data"@[SPACE]@[10xSPACE]@[8xSPACE]
      @[4xSPACE]@"
290   data"@@[6xSPACE]@[11xSPACE]@[4xSPA
      CE]@[SPACE]@"
300   data"@[2xSPACE]@[19xSPACE]@[3xSPAC
      E]@"
310   data"@[11xSPACE]@@@[12xSPACE]@"
320   data"@[4xSPACE]@[16xSPACE]@[4xSPAC
      E]@"
330   data"@[10xSPACE]@[5xSPACE]@[9xSPAC
      E]@"
340   data"@[3xSPACE]@@[8xSPACE]@[8xSPAC
      E]@@[2xSPACE]@"
350   data"@[10xSPACE]@[5xSPACE]@[9xSPAC
      E]@"
360   data"@b[CTRL-2]a[SPACE]a[SPACE]a[S
      PACE]a[SPACE]a[SPACE]a[SPACE]a[SPA
      CE]a[SPACE]a[SPACE]a[SPACE]a[SPACE
      ]a[SPACE] [COM-7]b@"
370   data"@[2xSPACE]@@@@@@@@@[8xSPACE]@
      [3xSPACE]@@[SPACE]@"
380   data"@[26xSPACE]@"
390   data"@[13xSPACE]@@@@@@@@@[4xSPACE]
      @"
400   data"@[5xSPACE]@@@@@[16xSPACE]@"
410   data"@[4xSPACE]@[5xSPACE]@[7xSPACE
      ]@[4xSPACE]@[2xSPACE]@"
420   data"@@@@[SPACE]@[SPACE]@@[4xSPACE
      ]@@@[3xSPACE]@[2xSPACE]@@[3xSPACE]
      @"
430   data"@[3xSPACE]@@@[2xSPACE]@@@@[3x
      SPACE]@@@@@@[2xSPACE]@@@@"
440   data"@[4xSPACE]@[11xSPACE]@[9xSPAC
      E]@"
450   data"@[SPACE]@[SPACE]@[2xSPACE]@@[
      2xSPACE]@[8xSPACE]@[2xSPACE]@[SPAC
      E]@[SPACE]@"
460   data"@[10xSPACE]@[3xSPACE]@[11xSPA
      CE]@"
470   data"b[CTRL-2]aaaa[CTRL-8]aaaaaaaa
      aaaa[CTRL-8]aaaaaaaaaa[COM-7]b"
480   data"b[CTRL-2] [SPACE]aaaa[8xSPACE]
      aaaaa[5xSPACE]a[COM-7]b@@"
490   data"@[8xSPACE]@@@@[14xSPACE]@"
500   data"@[13xSPACE]@@@@@@@@[2xSPACE]@
      @@@"
510   data"@@@[2xSPACE]@@@@@@[16xSPACE]@
      "
520   data170,101,101,170,89,89,170,101
530   data 0,0,0,0,255,0,0,0
540   data0,40,40,85,85,40,40,0
550   data0,20,0,0,85,0
560   data0,85,0,1,85,64
570   data1,85,64,1,85,64
580   data1,85,64,0,85,0
590   data0,215,0,0,195,0
600   data0,195,0,0,195,0
610   data0,195,0,0,195,0
620   data0,195,0,0,40,0
630   data0,40,0,0,40,0
640   data0,0,0,0,0,0
650   data0,0,0,0,20,0
660   data0,85,0,0,85,0
670   data1,85,64,1,85,64
680   data1,85,64,1,85,64
690   data0,85,0,0,215,0
700   data0,195,0,0,195,0
710   data0,195,0,0,195,0
720   data0,195,0,0,195,0
730   data0,40,0,0,43,0
740   data0,43,0,0,3,0
750   data0,51,48,0,15,192
760   fort=54272to54295:poket,0:next:pok
      e54296,15
770   poke54272,255 :poke54273,100:rem*1
      *
780   poke54277,100:poke54278,255
790   poke54280,10:poke54279,100:rem*2*
800   poke54284,10:poke54285,10
810   x=100:y=211:pokev+0,x:pokev+1,y
820   data8,0,32,8,60,32
830   data4,235,16,4,40,16
840   data1,40,64,1,85,64
850   data0,20,0,0,20,0
860   data0,20,0,0,20,0
870   data0,65,0,1,0,64
880   data1,0,64,1,0,64
890   data1,0,64,0,0,0
900   data0,0,0,0,0,0
910   data0,0,0,0,0,0,0,0,0
920   print"[HOME]";:y=211:pokev+1,y:la(
      1)=la(1)-1:goto 950
930   print"[HOME]";:y=211:pokev+1,y:la(
      1)=la(1)-1:ifla(1)=0then1320
940   sc=sc+100:goto1730
950   fort=1to18:printa$(int(8*rnd(1))+r
      s):next :pokev+31,0
960   fort=1to4:print"@@[24xSPACE]@@":ne
      xt:poke54276,129
970   ifint(la*rnd(1))=1thenpoke54283,12
      8:poke12300,255:poke54283,33
980   rem
990   ifint(30*rnd(1))=1then print"[HOME
      ]"b$  a$(int(15*rnd(1))):b$=b$+"[C
      RSR-DOWN]"
1000  if len(b$)>20thenb$="[2xCRSR-DOWN]
      "
1010  ifpeek(56320)=111thenpoke2040,14:p
      oke54276,128
1020  ifpeek(v+31)=1then1180
1030  ifint(50*rnd(1))=1thenpoke53265,in
      t(7*rnd(1))+24
1040  ifpeek(56320)=119thenpoke2040,13:p
      oke54276,129:goto 1120
1050  ifpeek(56320)=123thenpoke2040,13:p
      oke54276,129:goto1140
1060  ifpeek(56320)=125andpeek(v+1)<225
       thenpoke2040,13:poke54276,129:got
      o1160
1070  ifpeek(2040)=13theny=y-2:pokev+1,y
      +1:pokev+1,y:ify<47then930
1080  ifpeek(56320)=126thenpoke2040,13:p
      oke54276,129
1090  ifpeek(56320)=119thenpoke2040,13:p
```

```
     oke54276,129:goto 1120
1100 ifpeek(56320)=123thenpoke2040,13:p
     oke54276,129:goto1140
1110 poke12288+12,0:goto970
1120 pokev+0,x:pokev+0,x+1:pokev+0,x+2:
     pokev+0,x+3:pokev+0,x+4:pokev+0,x+
     5
1130 x=x+6:goto970
1140 pokev+0,x:pokev+0,x-1:pokev+0,x-2:
     pokev+0,x-3:pokev+0,x-4:pokev+0,x-
     5
1150 x=x-6:goto970
1160 pokev+1,y:pokev+1,y+1:pokev+1,y+2:
     pokev+1,y+3:pokev+1,y+4 :pokev+1,y
     +4
1170 y=y+6:goto970
1180 poke2040,13:sc=sc+(211-peek(v+1)):
     poke53283,0
1190 forq=1to250step10:poke54276,129:po
     ke54273,q:pokev+37,q+2:pokev+38,q+
     3
1200 pokev+1,255*rnd(1):pokev+0,250*rnd
     (1):pokev+23,2*rnd(1):pokev+29,2*r
     nd(1)
1210 poke53283,q:poke53282,q+4:pokev+39
     ,q+5:poke53265,int(7*rnd(1))+24:ne
     xt
1220 print"[SHIFT-CLR]":pokev+21,0:poke
     53281,0:poke53280,0:poke54296,0:po
     kev+23,0:
1230 pokev+37,2:pokev+39,2:pokev+38,2:f
     ort=832to832+62:poket,0:next:pokev
     +29,0
1240 print"[SHIFT-CLR][CTRL-8]":poke532
     72,21:poke53270,200:print"[3xCRSR-
     DOWN][SPACE]je[SPACE]score[SPACE]i
     s[SPACE]:"sc
1250 print"[3xCRSR-DOWN][SPACE]laatste[
     SPACE]score[SPACE]was:"ls
1260 print"[3xCRSR-DOWN][SPACE]gehaald[
     SPACE]door[SPACE]de[SPACE]kampioen
     [SPACE]"ls$:print"[3xCRSR-DOWN][SP
     ACE]";
1270 print"typ[SPACE]je[SPACE]naam[SPAC
     E]in[SPACE]a.u.b[SPACE]5[SPACE]let
     ters";:inputa$:iflen(a$)>5ora$=""t
     hen1240
1280 ls=sc:ls$=a$
1290 print"[3xCRSR-DOWN][14xSPACE][CTRL
      9]vuurknop
1300 ifpeek(56320)=111thengoto50
1310 goto1300
1320 pokev+21,3:print"[SHIFT-CLR]":poke
     v+0,100:pokev+2,100+11 :pokev+1,10
     0:pokev+3,113
1330 pokev+27,3:print"[CTRL-8][2xSPACE]
     bonus[SPACE]!!!![SPACE]..300.punte
     n..!!":sc=sc+300:poke54272,100
1340 poke53272,21:poke53270,200:print"[
     CTRL-8]"
1350 poke54276,33:
1360 pokev+1,peek(v+1)+4:pokev+3,peek(v
     +3)+4:
1370 ifpeek(v+3)>250thenpokev+21,1:poke
     v+3,10
1380 ifpeek(v+1)>250thengoto1420
1390 poke53280,peek(v+1)
1400 poke54273,int(200*rnd(1))+20
1410 goto1360
1420 la=la-1:ifla=3thenla=5
1430 print"[SHIFT-CLR][COM-7]":poke5427
     6,128:poke54272,255:poke54273,100:
     rs=rs+4:ifrs>22thenrs=21
1440 poke53272,(peek(53272)and240)+12
1450 poke53270,peek(53270)or16
1460 poke53280,0:la(1)=2 :x=100
1470 goto950
1480 fort=54272to54296:poket,0:poke5429
     6,15:poke54273,100:poke54272,100
1490 poke54277,100:poke54278,200:poke54
     276,33:s$="[SPACE]ballon-expeditie
     [2xSPACE]"
1500 print"[SHIFT-CLR]":fort=1to18step.
     1:poke54273,t:print"[HOME][CRSR-RI
     GHT][8xCRSR-DOWN][9xCRSR-RIGHT]"le
     ft$(s$,t)
1510 poke646,int(3*rnd(1))+2
1520 next:poke54276,32
1530 q$="[CRSR-RIGHT][8xCRSR-DOWN]":pok
     e54276,17:rem: r$="[CRSR-UP][9xCRS
     R-RIGHT][18xSPACE][CRSR-DOWN]"
1540 fort=1to500:next
1550 print"[SHIFT-CLR][CTRL-8]"
1560 print"[HOME][3xCRSR-DOWN]"
1570 print"
1580 print"[COM-M][COM-T]M[SPACE]NM[COM
      M][SPACE][COM-M][SPACE]NM[COM-M]M
     [SPACE][COM-G][15xSPACE]joystick
1590 print"[COM-M][SPACE]N[SPACE][COM-G
     ][2xCOM-M][SPACE][COM-M][SPACE][CO
     M G][3xCOM-M][SPACE][COM-G][18xSPA
     CE]in
1600 print"[COM-M][COM-T]M[SPACE]OP[COM
      M][SPACE][COM-M][SPACE][COM-G][3x
     COM-M][SPACE][COM-G]O[2xCOM-Y]P[12
     xSPACE]port[SPACE]2
1610 print"[COM-M][COM-@]N[SPACE][COM-G
     ][2xCOM-M][COM-@][COM-M][COM-@]MN[
     COM-M][SPACE]M[COM-G][4xCOM-T]
1620 print"N
1630 print"[COM-G][SPACE]MN[COM-M][COM-
     Y]M[COM-M][COM-Y]M[COM-M][COM-Y]MP
     O[COM-N][COM-P][SPACE]PO[COM-M][CO
     M Y]M
1640 print"O[SPACE][COM-M][COM-G][COM-M
     ][COM-P]N[COM-M][COM-P][SHIFT-@][C
     OM M][SPACE][2xCOM-M][COM-G][COM-N
     ][2xSPACE][COM-M][COM-G][COM-M][CO
     M P][SHIFT-@]
1650 print"M[SPACE]NM[COM-M][2xSPACE][C
     OM M][2xCOM-P][COM-M][COM-@]N[SHIF
     T @]L[COM-N][COM-P][SPACE][SHIFT-@
     ]L[COM-M][2xCOM-P]
1660 fort=15to0step-.1:poke54296,t:next
1670 print"[3xCRSR-DOWN][COM-6][CRSR-RI
     GHT][CTRL-9][15xSPACE][CTRL-0]
1680 print"[CRSR-RIGHT][CTRL-9][SPACE]d
     usty[2xSPACE]murray[SHIFT-SPACE][C
     TRL 0]
1690 print"[CRSR-RIGHT][CTRL-9][15xSPAC
     E][CTRL-0]
1700 print"[3xCRSR-DOWN][COM-1][SPACE]d
     ruk[SPACE]op[SPACE]de[SPACE]vuurkn
     op
1710 ifpeek(56320)=111then50
1720 goto1710
1730 pokev+21,3:print"[SHIFT-CLR]":poke
     v+0,120:pokev+2,120:pokev+1,11
     4:pokev+3,100
1740 pokev+27,3:print"[CTRL-8][2xSPACE]
     bonus[SPACE]!!!![SPACE]..100.punte
     n..!!":poke54272,100:poke54276,0
```

```
1750  poke53272,21:poke53270,200:fort=1t
      o1000:next:poke54276,33
1760  fort=1to22:pokev+3,peek(v+3)-1:pok
      e54273,int(255*rnd(1)):forz=1to50:
      next
1770  next:poke54276,129:fort=1to22:poke
      v+3,peek(v+3)+1:next:poke54273,0
1780  fort=1to50:pokev+3,peek(v+3)+2:po
      kev+1,peek(v+1)+2: next
1790  poke53272,(peek(53272)and240)+12
1800  poke53270,peek(53270)or16:print"[S
      HIFT CLR][COM-7]":poke54276,128:po
      ke54272,255:poke54273,100
1810  la=la-5:if la<7then 1840
1820  rs=rs+2:ifrs>25then rs=24
1830  x=100:y=211:pokev+3,250:pokev+0,x
      :goto 950
1840  poke53270,200:print"[SHIFT-CLR]"ta
      b(14)"[CTRL-1][9xCRSR-DOWN][CTRL-9
      ]hyperbonus":pokev+21,0
1850  poke 53272,21:poke53280,11
1860  print"[5xCRSR-DOWN][12xSPACE][CTRL
       9]einde[3xSPACE]spel"km(0)=0:km(2
      )=12:km(3)=15:km(4)=1
1870  print"[5xCRSR-DOWN][7xSPACE][CTRL-
      9]you[2xSPACE]score[2xSPACE]is[2xS
      PACE]1[SPACE]miljoen":km(1)=11:pok
      e54276,129
1880  printspc(13)"[5xCRSR-DOWN][CTRL-9]
      druk[2xSPACE]spatie":poke53272,21:
      poke53270,200:poke53265,91
1890  fort=0to4:poke53283,km(t):geta$:po
      ke54273,255*rnd(1):forz=1to100:nex
      t:next
1900  ifa$="[SPACE]"then run
1910  fort=4to0step-1:poke53283,km(t):ge
      ta$:poke54273,255*rnd(1)   :forz=1t
      o100
1920  next:next:goto 1890
```

** EINDE LISTING ballonex **

REGEL	10	27
REGEL	20	163
REGEL	30	136
REGEL	40	86
REGEL	50	229
REGEL	60	112
REGEL	70	225
REGEL	80	217
REGEL	90	133
REGEL	100	127
REGEL	110	224
REGEL	120	225
REGEL	130	206
REGEL	140	154
REGEL	150	165
REGEL	160	178
REGEL	170	199
REGEL	180	7
REGEL	190	201
REGEL	200	71
REGEL	210	239
REGEL	220	135
REGEL	230	71
REGEL	240	199
REGEL	250	71
REGEL	260	71
REGEL	270	71
REGEL	280	7
REGEL	290	71
REGEL	300	199
REGEL	310	7
REGEL	320	199
REGEL	330	199
REGEL	340	135
REGEL	350	199
REGEL	360	118
REGEL	370	71
REGEL	380	71
REGEL	390	135
REGEL	400	135
REGEL	410	71
REGEL	420	71
REGEL	430	71
REGEL	440	199
REGEL	450	71
REGEL	460	199
REGEL	470	192
REGEL	480	244
REGEL	490	71
REGEL	500	7
REGEL	510	71
REGEL	520	23
REGEL	530	163
REGEL	540	129
REGEL	550	238
REGEL	560	52
REGEL	570	111
REGEL	580	52
REGEL	590	86
REGEL	600	93
REGEL	610	93
REGEL	620	34
REGEL	630	231
REGEL	640	127
REGEL	650	177
REGEL	660	249
REGEL	670	111
REGEL	680	111
REGEL	690	36
REGEL	700	93
REGEL	710	93
REGEL	720	93
REGEL	730	234
REGEL	740	185
REGEL	750	147
REGEL	760	226
REGEL	770	68
REGEL	780	0
REGEL	790	15
REGEL	800	145
REGEL	810	128
REGEL	820	47
REGEL	830	147
REGEL	840	102
REGEL	850	227
REGEL	860	227
REGEL	870	245
REGEL	880	245
REGEL	890	186
REGEL	900	127
REGEL	910	147
REGEL	920	210
REGEL	930	148
REGEL	940	167
REGEL	950	17
REGEL	960	4
REGEL	970	237
REGEL	980	143
REGEL	990	127
REGEL	1000	61
REGEL	1010	24
REGEL	1020	86
REGEL	1030	103
REGEL	1040	167
REGEL	1050	164
REGEL	1060	231
REGEL	1070	202
REGEL	1080	30
REGEL	1090	167
REGEL	1100	164
REGEL	1110	104
REGEL	1120	53
REGEL	1130	165
REGEL	1140	58
REGEL	1150	166
REGEL	1160	64
REGEL	1170	167
REGEL	1180	181
REGEL	1190	24
REGEL	1200	38
REGEL	1210	214
REGEL	1220	98
REGEL	1230	26
REGEL	1240	194
REGEL	1250	94
REGEL	1260	79
REGEL	1270	165
REGEL	1280	251
REGEL	1290	138
REGEL	1300	120
REGEL	1310	77
REGEL	1320	17
REGEL	1330	155
REGEL	1340	110
REGEL	1350	107
REGEL	1360	164
REGEL	1370	168
REGEL	1380	14
REGEL	1390	9
REGEL	1400	85
REGEL	1410	83
REGEL	1420	250
REGEL	1430	206
REGEL	1440	127
REGEL	1450	239
REGEL	1460	247
REGEL	1470	39
REGEL	1480	76
REGEL	1490	142
REGEL	1500	177
REGEL	1510	97
REGEL	1520	236
REGEL	1530	30
REGEL	1540	173
REGEL	1550	14
REGEL	1560	35
REGEL	1570	187
REGEL	1580	223
REGEL	1590	160
REGEL	1600	130
REGEL	1610	246
REGEL	1620	137
REGEL	1630	62
REGEL	1640	170
REGEL	1650	82
REGEL	1660	139
REGEL	1670	72
REGEL	1680	149
REGEL	1690	124
REGEL	1700	87
REGEL	1710	239
REGEL	1720	82
REGEL	1730	10
REGEL	1740	121
REGEL	1750	55
REGEL	1760	92
REGEL	1770	92
REGEL	1780	29
REGEL	1790	127
REGEL	1800	3
REGEL	1810	92
REGEL	1820	190
REGEL	1830	33
REGEL	1840	79
REGEL	1850	138
REGEL	1860	234
REGEL	1870	161
REGEL	1880	210
REGEL	1890	58
REGEL	1900	23
REGEL	1910	71
REGEL	1920	211

Spreadsheet 128

E. Schefers uit s'Hertogenbosch stuurde ons een minisheet toe, dit is een (uitgebreide) minispreadsheet. In het programma is een duidelijke uitleg opgenomen. Omdat dit programma voor vele mensen interessant kan zijn hebben we het opgenomen ondanks de lengte.

```
100   rem ***********************
110   rem *      minisheet       *
120   rem *     -----------      *
130   rem *   commodore 128      *
```

```
140  rem * door erik scheffers *
150  rem * 1.0/251087         *
160  rem ***********************
170  :
180  dimcl$(6,19),ps%(6,19),br(6,19)
190  scnclr
200  cs$=chr$(142)+chr$(19)+chr$(19)+ch
     r$(147)
210  ifrgr(x)<5thengraphic0:slow:cs$=cs
     $+chr$(18):elsefast
220  printcs$tab(rgr(x)*4)"minisheet[12
     xSPACE]door[SPACE]erik[SPACE]schef
     fers"
230  print:printchr$(27)chr$(84)"moment
     je...":cc$="[10xSPACE]"
240  ifrgr(x)thenak=7:elseak=3
250  forx=0to6
260  k$=k$+"[CRSR-RIGHT][4xSPACE]"+chr$
     (65+x)+"[5xSPACE]"
270  fory=0to19
280  cl$(x,y)=cc$:poke5120+x+(y*7),0
290  next:next
300  fori=0to10:poke4096+i,0:next:rem f
     unctie-toetsen uit
310  poke4105,1:poke4106,72
320  cr$=chr$(27)+chr$(81)
330  ws$=chr$(19)+cr$+chr$(13)+cr$+chr$
     (13)+cr$+chr$(19)
340  ee$(1)="algemene[SPACE]fout"
350  ee$(2)="deling[SPACE]door[SPACE]nu
     l"
360  ee$(3)="berekening[SPACE]met[SPACE
     ]tekst[SPACE]of[SPACE]lege[SPACE]c
     el"
370  ee$(4)="te[SPACE]groot[SPACE]getal
     "
380  trap1330
390  gosub500
400  ifwr>0thenwr=wr-1:ifwr=0thenchar,0
     ,22,"[39xSPACE]"
410  gosub1430:ifso=2thengosub500:so=0
420  onkzgosub1550,1910,2160,2340,2510,
     2620,2910,3100,3290,3420
430  ifkz<11thenon(so+1)goto400,390,410
440  printws$;"einde"
450  print"dit[SPACE]wist[SPACE]alle[SP
     ACE]gegevens!"
460  print"wilt[SPACE]u[SPACE]echt[SPAC
     E]stoppen?[SPACE](j/n)"
470  getkeyi$:ifinstr("jn",i$)=0then470
480  ifi$="n"then400
490  printcs$;:end
500  scnclr:so=0
510  ifrgr(x)=0thenchar,2,3,(mid$(k$,1+
     (kx*11),34)),1
520  ifrgr(x)thenchar,2,3,k$,1
530  fori=1to9
540  char,0,3+(i*2),right$("0"+mid$(str
     $(i+ky),2),2),1
550  next:print"[HOME]"
560  rv=0:fory=0to8
570  forx=0toak-1
580  px=x:py=y:gosub620
590  next:next
600  return
610  rem --- cellen printen
620  char,3+(px*11),5+(py*2),cl$(px+kx,
     py+ky),rv
630  return
640  rem --- term halen
650  tb=0:ee=0:ne=0
660  i=asc(left$(tm$,1))
670  ifi=45thenne=1-ne:tm$=mid$(tm$,2):
     goto660
680  ifi=43ori=32thentm$=mid$(tm$,2):go
     to660
690  ifi=222ori=255thentm=:goto790
700  ifi>47andi<58then780
710  ifnot(i>64andi<72)thenee=1:return
720  kl=i-65:tm$=mid$(tm$,2):rg=val(tm$
     )-1
730  ifkl<0orkl>6orrg<0orrg>19thenee=1:
     return
740  ifleft$(cl$(kl,rg),1)="$"orcl$(kl,
     rg)=cc$thenee=3:return
750  iflen(tm$)>2orrg<0orrg=>20thenee=1
     :return
760  tm=br(kl,rg):ifpeek(5120+kl+(rg*7)
     )=0thentb=1
770  goto790
780  tm=val(tm$)
790  ifne=1thentm=-tm
800  ifright$(tm$,1)="%"thentm=tm/100
810  return
820  rem --- term berekenen
830  ifleft$(cl$,1)=">"thengosub1040:if
     eethenreturn
840  b$="+":u=0:tm$="":foria=1tolen(cl$
     )
850  z$=mid$(cl$,ia,1)
860  ifz$="+"orz$="-"orz$="*"orz$="/"or
     z$="[kwadraatpijl]"then900
870  ifz$=chr$(32)then890
880  tm$=tm$+z$
890  next:z$="":iftm$=""thenreturn
900  iftm$=""then880
910  gosub650:tm$="":ifeeortbthen1020
920  ifu>1e10thenee=4:goto1020
930  ifb$="+"thenu=u+tm
940  ifb$="-"thenu=u-tm
950  ifb$="*"thenu=u*tm
960  ifb$="/"andtm=0thenee=2:goto1020
970  ifb$="/"thenu=u/tm
980  ifb$="[kwadraatpijl]"and(tm>9oru>1
     0000)thenee=4:goto1020
990  ifb$="[kwadraatpijl]"thenu=u~tm
1000 ifz$=""thenreturn
1010 b$=z$:next
1020 ifia<len(cl$)thenia=len(cl$)+1:nex
     t
1030 return
1040 p$=mid$(cl$,2):cl$=""
1050 k1=asc(left$(p$,1))-65
1060 r1=val(mid$(p$,2)):p=3
1070 z$=mid$(p$,p,1)
1080 q=instr("+-*/[kwadraatpijl]",z$):i
     fq=0andp=3thenp=4:goto1070
1090  ifq=0ork1<0ork1>7orr1<1orr1>20then
     1150
1100 k2=asc(mid$(p$,p+1,1))-65
1110 r2=val(mid$(p$,p+2))
1120 ifk2<0ork2>7orr2<1orr2>20then1150
1130 ifk1=k2andr1<r2then1170
1140 ifk1<k2andr1=r2then1160
1150 ee=1:return
1160 r=r1:fork=k1tok2:gosub1190:next:go
     to1180
1170 k=k1:forr=r1tor2:gosub1190:next
1180 cl$=left$(cl$,len(cl$)-1):return
1190 cl$=cl$+chr$(k+65)+mid$(str$(r),2)
     +z$:return
1200 printws$"ik[SPACE]bereken[SPACE]nu
```

```
     [SPACE]alle[SPACE]cellen.[SPACE]mo
     mentje!":tr=0
1210 forx=0to6:fory=0to19
1220 cl$=cl$(x,y):ifpeek(5120+x+(y*7))=
     255thentr=tr+1:goto1290
1230 ifleft$(cl$,1)="$"orcl$=cc$thenbr(
     x,y)=0:tr=tr+1:goto1290
1240 gosub830
1250 ifeethenchar,0,22,(ee$(ee)+"[SPACE
     ]in[SPACE]"+chr$(65+x)+mid$(str$(y
     +1),2)):x=7:y=20:tr=140:wr=2:goto1
     290
1260 iftbthen1290
1270 br(x,y)=u
1280 poke5120+x+(y*7),255:tr=tr+1
1290 next:next
1300 iftr<140then1210
1310 return
1320 rem --- error opvang
1330 ifer>9then1420
1340 print"i/o[SPACE]error[SPACE]#";er:
     close2:close4
1350 print"(";err$(er);")"
1360 print"<return>[SPACE]om[SPACE]opni
     euw[SPACE]te[SPACE]proberen,"
1370 print"<m>[SPACE]voor[SPACE]terug[S
     PACE]naar[SPACE]menu"
1380 getkeyev$
1390 ifev$="m"thenresume390
1400 ifev$=chr$(13)thenresume
1410 goto1380
1420 print:printerr$(er)"[SPACE]in";el;
     :end
1430 rem --- commando halen
1440 printws$"commando[SPACE](i/r/l/s/g
     /p/v/w/h/m/e)[SPACE]?":px=cx:py=cy
1450 print"(help-toets[SPACE]voor[SPACE
     ]info)[SPACE]>[SPACE]";
1460 char,25,1,"*":ifso<2thenrv=1:gosub
     620
1470 fori=1toak*10:geti$:ifi$<>""theni=
     71:next:goto1500:elsenext
1480 char,25,1,".":ifso<2thenrv=0:gosub
     620
1490 fori=1toak*10:geti$:ifi$<>""theni=
     71:next:elsenext:goto1460
1500 kz=instr("irlsgpvwmhe",i$)
1510 ifkz=0then1460
1520 char,25,1,i$:ifso<2thenrv=0:gosub6
     20
1530 return
1540 rem --- invoeren
1550 printws$"invoer"
1560 print"cursor[SPACE]besturen[SPACE]
     met[SPACE]crsr[SPACE]toetsen"
1570 print"overige[SPACE]toetsen=[SPACE
     ]invoer;[SPACE]return[SPACE]=[SPAC
     E]menu"
1580 ps%(tx,ty)=ps:px=cx:py=cy:tx=cx+kx
     :ty=cy+ky:ps=ps%(tx,ty)
1590 rv=1:gosub620
1600 fori=1toak*10:geti$:ifi$<>""theni=
     71:next:goto1630:elsenext
1610 rv=0:gosub620
1620 fori=1toak*10:geti$:ifi$<>""theni=
     71:next:elsenext:goto1590
1630 rv=0:gosub620
1640 ifi$="[CRSR-UP]"then1770
1650 ifi$="[CRSR-DOWN]"then1800
1660 ifi$="[CRSR-LEFT]"then1830
1670 ifi$="[CRSR-RIGHT]"then1860
```

```
1680 ifi$="[SHIFT-CLR]"thencl$(tx,ty)=c
     c$:ps=0:goto1580
1690 ifi$="[HOME]"thencx=0:kx=0:cy=0:ky
     =0:gosub510:goto1580
1700 ifi$=chr$(13)thenreturn
1710 ifi$=chr$(20)andps=0then1590
1720 ifi$=chr$(20)thenmid$(cl$(tx,ty),p
     s)="[SPACE]":ps=ps-1:goto1590
1730 i=asc(i$):ifi=222then1760:rem pi d
     oorlaten
1740 ifi<32ori=34ori>94orps=10then1590
1750 ifi=44ori=58ori=59then1590
1760 ps=ps+1:mid$(cl$(tx,ty),ps)=i$:got
     o1590
1770 ifcy=0andky=0then1580
1780 ifcy=0thenky=ky-1:gosub530:goto158
     0
1790 cy=cy-1:goto1580
1800 ifcy=8andky=11then1580
1810 ifcy=8thenky=ky+1:gosub530:goto158
     0
1820 cy=cy+1:goto1580
1830 ifcx=0andkx=0then1580
1840 ifcx=0thenkx=kx-1:gosub510:goto158
     0
1850 cx=cx-1:goto1580
1860 ifrgr(x)=0andcx=2andkx=4then1580
1870 ifrgr(x)=0andcx=2thenkx=kx+1:gosub
     510:goto1580
1880 ifrgr(x)=5andcx=6then1580
1890 cx=cx+1:goto1580
1900 rem --- rekenen
1910 gosub1200:ifeethen2130
1920 scnclr:sx=kx:sy=ky
1930 ifrgr(x)=0thenchar,2,3,(mid$(k$,1+
     (sx*11),34)),1
1940 ifrgr(x)thenchar,2,3,k$,1
1950 fori=1to9
1960 char,0,3+(i*2),right$("0"+mid$(str
     $(i+sy),2),2),1
1970 next
1980 fory=0to8
1990 forx=0toak-1
2000 char,3+(x*11),5+(y*2)
2010 ifpeek(5120+x+sx+((y+sy)*7))=255th
     enprintusing"######.##-";br(x+sx,y
     +sy):elseprintmid$(cl$(x+sx,y+sy),
     2)"[SPACE]"
2020 next:next
2030 print"[HOME]gebruik[SPACE]de[SPACE
     ]crsr[SPACE]toetsen[SPACE]om[SPACE
     ]te[SPACE]scrollen"
2040 print"return[SPACE]voor[SPACE]teru
     gkeer[SPACE]naar[SPACE]werkblad."
2050 getkeyi$
2060 ifi$=chr$(13)then2120
2070 ifi$="[CRSR-UP]"andsy>0thensy=sy-1
     :goto1930
2080 ifi$="[CRSR-DOWN]"andsy<11thensy=s
     y+1:goto1930
2090 ifi$="[CRSR-LEFT]"andsx>0andrgr(x)
     =0thensx=sx-1:goto1930
2100 ifi$="[CRSR-RIGHT]"andsx<4andrgr(x
     )=0thensx=sx+1:goto1930
2110 goto2050
2120 so=1
2130 fori=0to139:poke5120+i,0:next
2140 return
2150 rem --- inladen
2160 scnclr:so=1
2170 print"[HOME]inladen"
```

```
2180 print:print"filenaam[SPACE]($[SPACE]directory/return[SPACE]menu)"
2190 fl$="":inputfl$
2200 iffl$=""thenreturn
2210 iffl$="$"thenscnclr:directory:goto2180
2220 iflen(fl$)>16then2190
2230 print:print"(d)isk[SPACE]of[SPACE](t)ape"
2240 getkeyi$:ifinstr("dt",i$)=0then2240
2250 ifi$="d"thendv=8:su=2:fl$=fl$+",s,r":elsedv=1:su=0
2260 open2,dv,su,fl$
2270 forx=0to6:fory=0to19
2280 input#2,cl$:input#2,ps%(x,y)
2290 ifcl$=chr$(2)thencl$=cc$:elsecl$=left$(cl$,10)
2300 cl$(x,y)=cl$
2310 next:next
2320 close2:kx=0:ky=0:cx=0:cy=0:return
2330 rem --- opslaan
2340 scnclr:so=1
2350 print"[HOME]opslaan"
2360 print:print"filenaam[SPACE]($[SPACE]directory/return[SPACE]menu)"
2370 fl$="":inputfl$
2380 iffl$=""thenreturn
2390 iffl$="$"thenscnclr:directory:goto2180
2400 iflen(fl$)>16then2190
2410 print:print"(d)isk[SPACE]of[SPACE](t)ape"
2420 getkeyi$:ifinstr("dt",i$)=0then2240
2430 ifi$="d"thendv=8:su=2:fl$="@:"+fl$+",s,w":elsedv=1:su=1
2440 open2,dv,su,fl$
2450 forx=0to6:fory=0to19
2460 cl$=cl$(x,y):ifcl$=cc$thencl$=chr$(2):elsecl$=cl$+chr$(2)
2470 print#2,cl$:print#2,ps%(x,y)
2480 next:next
2490 close2:kx=0:ky=0:cx=0:cy=0:return
2500 rem --- ga naar
2510 printws$;"ga[SPACE]naar"
2520 input"naar[SPACE]welke[SPACE]cel[SPACE]springen";c$:so=1
2530 kl=asc(left$(c$,1))-65:rg=val(mid$(c$,2))-1
2540 ifkl<0orkl>6orrg<0orrg>19then2510
2550 ifkl<akthenkx=0:cx=kl:goto2580
2560 ifkl>4thenkx=4:cx=kl-4:goto2580
2570 kx=kl:cx=0
2580 ifrg<9thenky=0:cy=rg:return
2590 ifrg>11thenky=11:cy=rg-11:return
2600 ky=rg:cy=0:return
2610 rem --- printen
2620 printws$;"printen"
2630 print"(w)erkveld[SPACE]/[SPACE](u)itkomsten[SPACE](return=menu)"
2640 getkeyi$:ifi$=chr$(13)thenreturn
2650 k=instr("uw",i$)
2660 ifk=0then2640
2670 ifk=1thengosub1200:ifeethenreturn
2680 printws$;"printen"
2690 print"printer[SPACE]inschakelen."
2700 print"<toets>":getkeyi$:so=1:scnclr
2710 open4,4
2720 print#4,"[8xSPACE]a[10xSPACE]b[10xSPACE]c[10xSPACE]d[10xSPACE]e[10xSPACE]f[10xSPACE]g"
2730 onkgoto2800
2740 forr=0to19
2750 print#4,right$("0"+mid$(str$(r+1),2),2);"[SPACE]";
2760 fork=0to6
2770 print#4,cl$(k,r);"[SPACE]";
2780 next:print#4:next
2790 close4:return
2800 cmd4
2810 forr=0to19
2820 printright$("0"+mid$(str$(r+1),2),2);"[SPACE]";
2830 fork=0to6
2840 ifpeek(5120+k+(r*7))then2860
2850 printmid$(cl$(k,r),2);"[2xSPACE]";:goto2870
2860 printusing"######.##-";br(k,r);:print"[SPACE]";
2870 next:print:next
2880 print#4:close4
2890 return
2900 rem --- invoegen
2910 printws$"invoegen"
2920 print"(r)egel[SPACE]of[SPACE](k)olom/<return>[SPACE]=[SPACE]menu"
2930 getkeyi$:ifi$=chr$(13)thenreturn
2940 ifinstr("rk",i$)=0then2930
2950 printws$"invoegen"
2960 print"bij[SPACE]welke[SPACE]regel/kolom[SPACE]invoegen?"
2970 inputiv$:so=1
2980 ifi$="k"then3040
2990 r=val(iv$)-1:ifr<0orr>19then2950
3000 ifr=19then3030
3010 fori=18torstep-1
3020 fork=0to6:cl$(k,i+1)=cl$(k,i):next:next
3030 fork=0to6:cl$(k,r)=cc$:next:return
3040 k=asc(iv$)-65:ifk<0ork>6then2950
3050 ifk=6then3080
3060 fori=5tokstep-1
3070 forr=0to19:cl$(i+1,r)=cl$(i,r):next:next
3080 forr=0to19:cl$(k,r)=cc$:next:return
3090 rem --- wissen
3100 printws$"wissen"
3110 print"(r)egel[SPACE]of[SPACE](k)olom/<return>[SPACE]=[SPACE]menu"
3120 getkeyi$:ifi$=chr$(13)thenreturn
3130 ifinstr("rk",i$)=0then3120
3140 printws$"wissen"
3150 print"welke[SPACE]regel/kolom[SPACE]wissen?"
3160 inputiv$:so=1
3170 ifi$="k"then3230
3180 r=val(iv$)-1:ifr<0orr>19then3140
3190 ifr=19then3220
3200 fori=rto18
3210 fork=0to6:cl$(k,i)=cl$(k,i+1):next:next
3220 fork=0to6:cl$(k,19)=cc$:next:return
3230 r=asc(iv$)-65:ifr<0orr>6then3140
3240 ifk=6then3270
3250 fori=kto5
3260 forr=0to19:cl$(i,r)=cl$(i+1,r):next:next
3270 forr=0to19:cl$(6,r)=cc$:next:retur
```

```
     n
3280 rem --- menu
3290 restore3330
3300 fori=0to7:readt$
3310 char,ak*3,i+10,t$,1:next
3320 so=2:return
3330 data"[COM-A][11xSHIFT-*][COM-R][12
     xSHIFT-*][COM-S]
3340 data"[SHIFT--]i[SPACE]=[SPACE]invo
     er[SPACE][SHIFT--]v[SPACE]=[SPACE]
     invoegen[SHIFT--]
3350 data"[SHIFT--]r[SPACE]=[SPACE]reke
     nen[SHIFT--]w[SPACE]=[SPACE]wissen
     [2xSPACE][SHIFT--]
3360 data"[SHIFT--]l[SPACE]=[SPACE]lade
     n[2xSPACE][SHIFT--]h[SPACE]=[SPACE
     ]info[4xSPACE][SHIFT--]
3370 data"[SHIFT--]s[SPACE]=[SPACE]opsl
     aan[SHIFT--]m[SPACE]=[SPACE]menu[4
     xSPACE][SHIFT--]
3380 data"[SHIFT--]g[SPACE]=[SPACE]ga[S
     PACE]naar[SHIFT--]e[SPACE]=[SPACE]
     einde[3xSPACE][SHIFT--]
3390 data"[SHIFT--]p[SPACE]=[SPACE]prin
     t[2xSPACE][SHIFT--][12xSPACE][SHIF
     T -]
3400 data"[COM-Z][11xSHIFT-*][COM-E][12
     xSHIFT-*][COM-X]
3410 rem --- help
3420 scnclr
3430 printtab(17)"info"
3440 print
3450 print"extra[SPACE]hulp[SPACE]in[SP
     ACE]het[SPACE]hoofd-'menu'[SPACE]k
     unt[SPACE]u"
3460 print"krijgen[SPACE]door[SPACE]op[
     SPACE]'m'[SPACE]te[SPACE]drukken."
3470 print
3480 print"bij[SPACE]'invoeren'[SPACE]k
     unt[SPACE]u[SPACE]het[SPACE]werkbl
     ad"
3490 print"invullen.[SPACE]de[SPACE]vol
     gende[SPACE]invoer[SPACE]op[SPACE]
     het"
3500 print"werkblad[SPACE]is[SPACE]moge
     lijk:"
3510 print"[SPACE][CTRL-9]$tekst[2xSPAC
     E][CTRL-0][SPACE]na[SPACE]een[SPAC
     E]dollar[SPACE]teken[SPACE]kunt[SP
     ACE]u"
3520 print"[10xSPACE]tekst[SPACE]invoer
     en."
3530 print"[SPACE][CTRL-9]12[6xSPACE][C
     TRL 0][SPACE]de[SPACE]cel[SPACE]be
     vat[SPACE]nu[SPACE]het[SPACE]getal
     [SPACE]12"
3540 print"[SPACE][CTRL-9]2+2[5xSPACE][
     CTRL-0][SPACE]een[SPACE]berekening
     ."
3550 print"[12xSPACE]mogelijk[SPACE]is[
     SPACE]+,-,*,/[SPACE]en[SPACE][kwad
     raatpijl]."
3560 print"[SPACE][CTRL-9]25%[5xSPACE][
     CTRL-0][SPACE]procenten[SPACE]zijn
     [SPACE]ook[SPACE]mogelijk!"
3570 print
3580 print"u[SPACE]kunt[SPACE]natuurlij
     k[SPACE]ook[SPACE]met[SPACE]andere
     [SPACE]cellen"
3590 print"rekenen.[SPACE]dit[SPACE]doe
     t[SPACE]u[SPACE]door[SPACE]eerst[S
     PACE]de"
3600 print"kolom[SPACE]letter[SPACE]en[
     SPACE]dan[SPACE]het[SPACE]regel[SP
     ACE]nummer[SPACE]te"
3610 print"geven."
3620 print"<toets>";:getkeyi$
3630 scnclr
3640 print"hiermee[SPACE]kunt[SPACE]u[S
     PACE]ook[SPACE]rekenen,[SPACE]dus[
     SPACE]bv."
3650 print"[CTRL-9][SPACE]a2*3[SPACE][C
     TRL 0],[CTRL-9][SPACE]2[kwadraatpi
     jl]a5[SPACE][CTRL-0][SPACE]of[SPAC
     E][CTRL-9][SPACE]b6*b19[SPACE][CTR
     L 0]."
3660 print"het[SPACE]getal[SPACE][COM-P
     I][SPACE](pi,[SPACE]3.14159265)[SP
     ACE]is[SPACE]ook[SPACE]te"
3670 print"gebruiken!"
3680 print
3690 print"[SPACE][CTRL-9]>d2+d8[2xSPAC
     E][CTRL-0][SPACE]telt[SPACE]alles[
     SPACE]van[SPACE]d2[SPACE]t/m[SPACE
     ]d8[SPACE]op.":print
3700 print"[SPACE][CTRL-9]>a3*f3[2xSPAC
     E][CTRL-0][SPACE]vermenigvuldigd[S
     PACE]a3[SPACE]t/m[SPACE]f3."
3710 print
3720 print"'rekenen'[SPACE]hier[SPACE]w
     ordt[SPACE]uw[SPACE]werkblad[SPACE
     ]uit-"
3730 print"gerekend.[SPACE]dit[SPACE]du
     urt[SPACE]even.[SPACE]als[SPACE]er
     [SPACE]een"
3740 print"fout[SPACE]is,[SPACE]wordt[S
     PACE]die[SPACE]onderaan[SPACE]het[
     SPACE]scherm"
3750 print"gegeven,[SPACE]en[SPACE]spri
     ngt[SPACE]het[SPACE]programma[SPAC
     E]weer"
3760 print"naar[SPACE]het[SPACE]'menu'.
     [SPACE]als[SPACE]er[SPACE]geen[SPA
     CE]fouten[SPACE]zijn"
3770 print"worden[SPACE]de[SPACE]uitkom
     sten[SPACE]op[SPACE]het[SPACE]sche
     rm"
3780 print"getoond.[SPACE]scrollen[SPAC
     E]gaat[SPACE]met[SPACE]de[SPACE]cu
     rsor"
3790 print"toetsen."
3800 print
3810 print"'laden'[SPACE]spreekt[SPACE]
     voor[SPACE]zichzelf.[SPACE]u[SPACE
     ]kunt"
3820 print"kiezen[SPACE]uit[SPACE]diske
     tte[SPACE]of[SPACE]cassette."
3830 print:print"<toets>":getkeyi$:scnc
     lr
3840 print"'opslaan'[SPACE]hetzelfde[SP
     ACE]als[SPACE]laden."
3850 print
3860 print"'ga[SPACE]naar'[SPACE]hier[S
     PACE]kunt[SPACE]u[SPACE]een[SPACE]
     cel[SPACE]nummer"
3870 print"geven[SPACE]waar[SPACE]de[SP
     ACE]cursor[SPACE]naar[SPACE]toe[SP
     ACE]moet"
3880 print"springen."
3890 print
3900 print"'print'[SPACE]u[SPACE]kunt[S
     PACE]kiezen[SPACE]uit[SPACE]het[SP
```

```
     ACE]werkblad"
3910 print"of[SPACE]de[SPACE]uitkomsten
     ."
3920 print
3930 print"'invoegen'[SPACE]om[SPACE]ee
     n[SPACE]regel[SPACE]of[SPACE]kolom
     [SPACE]in[SPACE]te"
3940 print"voegen."
3950 print
3960 print"'wissen'[SPACE]wist[SPACE]ee
     n[SPACE]regel[SPACE]of[SPACE]kolom
     ."
3970 print
3980 print"'help'[SPACE]dit[SPACE]dus."
3990 print
4000 print"'menu'[SPACE]dit[SPACE]geeft
     [SPACE]alle[SPACE]commando's[SPACE
     ]met"
4010 print"hun[SPACE]keuze[SPACE]letter
     ."
4020 print
4030 print"'einde'[SPACE]om[SPACE]het[S
     PACE]programma[SPACE]te[SPACE]verl
     aten."
4040 print:print"<toets>":getkeyi$:scnc
     lr
4050 print"indien[SPACE]u[SPACE]beschik
     t[SPACE]over[SPACE]een[SPACE]compi
     ler,"


4070 print"compileren.[SPACE]dit[SPACE]
     scheelt[SPACE]met[SPACE]de"
4080 print"verwerkings[SPACE]snelheid."
4090 print
4100 print"dit[SPACE]programma[SPACE]ku
     nt[SPACE]u[SPACE]het[SPACE]beste[S
     PACE]op[SPACE]het"
4110 print"80[SPACE]koloms[SPACE]scherm
     [SPACE]draaien.[SPACE]40[SPACE]kol
     omen"
4120 print"gaat[SPACE]ook,[SPACE]maar[S
     PACE]is[SPACE]veel[SPACE]langzamer
     ."
4130 print"(dit[SPACE]komt[SPACE]doorda
     t[SPACE]de[SPACE]computer[SPACE]op
     [SPACE]de"
4140 print"80[SPACE]koloms[SPACE]scherm
     [SPACE]in[SPACE]de[SPACE]2[SPACE]m
     hz[SPACE](fast)[SPACE]mode"
4150 print"kan[SPACE]werken.)"
4160 print:print"<toets>":getkeyi$:so=1
     :return
4170 by esoft
```

** EINDE LISTING minishee **

REGEL	100	85
REGEL	110	137
REGEL	120	210
REGEL	130	35
REGEL	140	219
REGEL	150	216
REGEL	160	85
REGEL	170	58
REGEL	180	68
REGEL	190	232
REGEL	200	209
REGEL	210	184
REGEL	220	86
REGEL	230	120
REGEL	240	156
REGEL	250	149
REGEL	260	184
REGEL	270	202
REGEL	280	107
REGEL	290	62
REGEL	300	120
REGEL	310	239
REGEL	320	23
REGEL	330	163
REGEL	340	162
REGEL	350	253
REGEL	360	249
REGEL	370	186
REGEL	380	158
REGEL	390	34
REGEL	400	207
REGEL	410	39
REGEL	420	48
REGEL	430	241
REGEL	440	75
REGEL	450	152
REGEL	460	22
REGEL	470	138
REGEL	480	119
REGEL	490	72
REGEL	500	166
REGEL	510	38
REGEL	520	60
REGEL	530	138
REGEL	540	1
REGEL	550	172
REGEL	560	92
REGEL	570	199
REGEL	580	255
REGEL	590	62
REGEL	600	142
REGEL	610	233
REGEL	620	239
REGEL	630	142
REGEL	640	182
REGEL	650	205
REGEL	660	77
REGEL	670	152
REGEL	680	184
REGEL	690	191
REGEL	700	78
REGEL	710	216
REGEL	720	211
REGEL	730	159
REGEL	740	197
REGEL	750	93
REGEL	760	238
REGEL	770	41
REGEL	780	46
REGEL	790	71
REGEL	800	187
REGEL	810	142
REGEL	820	221
REGEL	830	101
REGEL	840	128
REGEL	850	17
REGEL	860	182
REGEL	870	128
REGEL	880	100
REGEL	890	229
REGEL	900	141
REGEL	910	28
REGEL	920	5
REGEL	930	96
REGEL	940	99
REGEL	950	97
REGEL	960	227
REGEL	970	103
REGEL	980	236
REGEL	990	111
REGEL	1000	52
REGEL	1010	82
REGEL	1020	208
REGEL	1030	142
REGEL	1040	53
REGEL	1050	69
REGEL	1060	167
REGEL	1070	152
REGEL	1080	65
REGEL	1090	172
REGEL	1100	159
REGEL	1110	51
REGEL	1120	205
REGEL	1130	15
REGEL	1140	14
REGEL	1150	53
REGEL	1160	183
REGEL	1170	49
REGEL	1180	200
REGEL	1190	180
REGEL	1200	31
REGEL	1210	153
REGEL	1220	124
REGEL	1230	122
REGEL	1240	40
REGEL	1250	77
REGEL	1260	148
REGEL	1270	201
REGEL	1280	115
REGEL	1290	62
REGEL	1300	228
REGEL	1310	142
REGEL	1320	107
REGEL	1330	122
REGEL	1340	61
REGEL	1350	163
REGEL	1360	28
REGEL	1370	200
REGEL	1380	89
REGEL	1390	166
REGEL	1400	245
REGEL	1410	85
REGEL	1420	195
REGEL	1430	204
REGEL	1440	23
REGEL	1450	254
REGEL	1460	71
REGEL	1470	236
REGEL	1480	74
REGEL	1490	241
REGEL	1500	177
REGEL	1510	132
REGEL	1520	69
REGEL	1530	142
REGEL	1540	124
REGEL	1550	126
REGEL	1560	101
REGEL	1570	173
REGEL	1580	131
REGEL	1590	234
REGEL	1600	240
REGEL	1610	233
REGEL	1620	245
REGEL	1630	233
REGEL	1640	245
REGEL	1650	111
REGEL	1660	254
REGEL	1670	129
REGEL	1680	93
REGEL	1690	74
REGEL	1700	91
REGEL	1710	206
REGEL	1720	212
REGEL	1730	84
REGEL	1740	241
REGEL	1750	149
REGEL	1760	49
REGEL	1770	179
REGEL	1780	118
REGEL	1790	87
REGEL	1800	237
REGEL	1810	125
REGEL	1820	86
REGEL	1830	177
REGEL	1840	113
REGEL	1850	85
REGEL	1860	189
REGEL	1870	120
REGEL	1880	142
REGEL	1890	84
REGEL	1900	30
REGEL	1910	12
REGEL	1920	94
REGEL	1930	46
REGEL	1940	60
REGEL	1950	138
REGEL	1960	9
REGEL	1970	130

```
REGEL 1980   152
REGEL 1990   199
REGEL 2000   51
REGEL 2010   116
REGEL 2020   62
REGEL 2030   117
REGEL 2040   77
REGEL 2050   7
REGEL 2060   146
REGEL 2070   216
REGEL 2080   139
REGEL 2090   231
REGEL 2100   108
REGEL 2110   80
REGEL 2120   133
REGEL 2130   145
REGEL 2140   142
REGEL 2150   17
REGEL 2160   167
REGEL 2170   235
REGEL 2180   91
REGEL 2190   33
REGEL 2200   108
REGEL 2210   160
REGEL 2220   224
REGEL 2230   60
REGEL 2240   183
REGEL 2250   182
REGEL 2260   77
REGEL 2270   153
REGEL 2280   167
REGEL 2290   189
REGEL 2300   70
REGEL 2310   62
REGEL 2320   136
REGEL 2330   36
REGEL 2340   167
REGEL 2350   254
REGEL 2360   91
REGEL 2370   33
REGEL 2380   108
REGEL 2390   160
REGEL 2400   224
REGEL 2410   60
REGEL 2420   183
REGEL 2430   36
REGEL 2440   77
REGEL 2450   153
REGEL 2460   139
REGEL 2470   207
REGEL 2480   62
REGEL 2490   136
REGEL 2500   192
REGEL 2510   144
REGEL 2520   254
REGEL 2530   170
REGEL 2540   50
REGEL 2550   61
REGEL 2560   198
REGEL 2570   163
REGEL 2580   38
REGEL 2590   140
REGEL 2600   111
REGEL 2610   54
REGEL 2620   6
REGEL 2630   200
REGEL 2640   156
REGEL 2650   171
REGEL 2660   43
REGEL 2670   52
REGEL 2680   6
REGEL 2690   84
REGEL 2700   8
REGEL 2710   51
REGEL 2720   24
REGEL 2730   47
REGEL 2740   195
REGEL 2750   3
REGEL 2760   136
REGEL 2770   127
REGEL 2780   68
REGEL 2790   156
REGEL 2800   209
REGEL 2810   195
REGEL 2820   164
REGEL 2830   136
REGEL 2840   2
REGEL 2850   45
REGEL 2860   193
REGEL 2870   17
REGEL 2880   218
REGEL 2890   142
REGEL 2900   113
REGEL 2910   6
REGEL 2920   0
REGEL 2930   156
REGEL 2940   129
REGEL 2950   6
REGEL 2960   228
REGEL 2970   7
REGEL 2980   167
REGEL 2990   71
REGEL 3000   102
REGEL 3010   96
REGEL 3020   79
REGEL 3030   111
REGEL 3040   57
REGEL 3050   48
REGEL 3060   37
REGEL 3070   152
REGEL 3080   170
REGEL 3090   239
REGEL 3100   132
REGEL 3110   0
REGEL 3120   156
REGEL 3130   121
REGEL 3140   132
REGEL 3150   141
REGEL 3160   7
REGEL 3170   168
REGEL 3180   63
REGEL 3190   103
REGEL 3200   219
REGEL 3210   79
REGEL 3220   135
REGEL 3230   70
REGEL 3240   49
REGEL 3250   160
REGEL 3260   152
REGEL 3270   149
REGEL 3280   75
REGEL 3290   85
REGEL 3300   192
REGEL 3310   180
REGEL 3320   78
REGEL 3330   245
REGEL 3340   131
REGEL 3350   64
REGEL 3360   218
REGEL 3370   153
REGEL 3380   81
REGEL 3390   86
REGEL 3400   0
REGEL 3410   63
REGEL 3420   232
REGEL 3430   61
REGEL 3440   153
REGEL 3450   201
REGEL 3460   48
REGEL 3470   153
REGEL 3480   42
REGEL 3490   168
REGEL 3500   81
REGEL 3510   99
REGEL 3520   252
REGEL 3530   7
REGEL 3540   240
REGEL 3550   247
REGEL 3560   82
REGEL 3570   153
REGEL 3580   174
REGEL 3590   181
REGEL 3600   82
REGEL 3610   128
REGEL 3620   98
REGEL 3630   232
REGEL 3640   66
REGEL 3650   180
REGEL 3660   96
REGEL 3670   154
REGEL 3680   153
REGEL 3690   205
REGEL 3700   45
REGEL 3710   153
REGEL 3720   2
REGEL 3730   112
REGEL 3740   48
REGEL 3750   126
REGEL 3760   51
REGEL 3770   138
REGEL 3780   231
REGEL 3790   45
REGEL 3800   153
REGEL 3810   23
REGEL 3820   17
REGEL 3830   28
REGEL 3840   70
REGEL 3850   153
REGEL 3860   20
REGEL 3870   35
REGEL 3880   113
REGEL 3890   153
REGEL 3900   52
REGEL 3910   60
REGEL 3920   153
REGEL 3930   176
REGEL 3940   207
REGEL 3950   153
REGEL 3960   215
REGEL 3970   153
REGEL 3980   79
REGEL 3990   153
REGEL 4000   120
REGEL 4010   74
REGEL 4020   153
REGEL 4030   219
REGEL 4040   28
REGEL 4050   209
REGEL 4060   79
REGEL 4070   81
REGEL 4080   174
REGEL 4090   153
REGEL 4100   207
REGEL 4110   119
REGEL 4120   199
REGEL 4130   78
REGEL 4140   193
REGEL 4150   218
REGEL 4160   129
REGEL 4170   28
```

Syntax Checksum C16

Evenals voor de C 64, is er ook voor de C 16 een aparte Checksum ontworpen. Hieronder dit handige programmeersteuntje voor de C 16.

```
10    rem ******************************
      ***
20    rem syntax.checksum
30    rem voor c-16 & plus/4
40    rem
50    rem syntax testen met 'sys 1536'
60    rem
70    rem v.851128.16       jan bodzinga
80    rem ******************************
      ***
90    i=1536:rem beginadres
100   reada:ifa>=0then pokei,a:i=i+1:got
      o100
110   print"data[SPACE]is[SPACE]weggezet
      "
120   print"cheksum[SPACE]printen[SPACE]
      met[SPACE]'sys[SPACE]1536'
130   end
200   data 165, 43,166, 44,133
210   data  31,134, 32,169,147
220   data  32,210,255,160,  0
230   data 240,  3, 32, 73,  6
240   data  32, 73,  6,208,  1
250   data  96, 72,152, 32,131
260   data   6,168,104,234, 32
270   data  81,  6, 32, 73,  6
280   data 240, 12,201, 32,240
290   data 247, 24,101,252,133
300   data 252, 76, 37,  6,166
310   data 252,169,  0,132,253
320   data  32, 95,164,169, 13
330   data  32,210,255,164,253
340   data  76, 17,  6,200,208
350   data   2,230, 32,177, 31
360   data  96,162,  0,189,123
370   data   6,240,  6, 32,210
380   data 255,232,208,245, 32
390   data  73,  6,170, 32, 73
400   data   6,132,253, 32, 95
410   data 164,162,  3,169, 32
420   data  32,210,255,202,208
430   data 250,169,  0,133,252
440   data 164,253, 96, 82, 69
450   data  71, 69, 76, 32,  0
460   data   0, 72,138, 72, 32
470   data 225,255,240,251,104
480   data 170,104, 96, -1
```

** EINDE LISTING checks16

REGEL 10	249	REGEL 250	157	REGEL 400	108
REGEL 20	247	REGEL 260	155	REGEL 410	159
REGEL 30	121	REGEL 270	215	REGEL 420	245
REGEL 40	143	REGEL 280	186	REGEL 430	202
REGEL 50	75	REGEL 290	248	REGEL 440	176
REGEL 60	143	REGEL 300	118	REGEL 450	12
REGEL 70	8	REGEL 310	204	REGEL 460	54
REGEL 80	249	REGEL 320	165	REGEL 470	43
REGEL 90	103	REGEL 330	252	REGEL 480	1
REGEL 100	2	REGEL 340	106		
REGEL 110	245	REGEL 350	98		
REGEL 120	237	REGEL 360	163		
REGEL 130	128	REGEL 370	45		
REGEL 200	210	REGEL 380	0		
REGEL 210	208	REGEL 390	58		
REGEL 220	142				
REGEL 230	1				
REGEL 240	3				

Klinkers

Js vn n s d mkr vn dt spl.
Zo'n soort zin zou voor kunnen komen bij het spelen van het spel zonder klinkers. (Er staat: Jos van Oijen is de maker van dit spel). In de woorden die er op het scherm verschijnen ontbreken alle klinkers, en dit is moeilijker dan u denkt. De woordenlijst is zelf uit te breiden.

```
10    printchr$(14)chr$(8):fora=1to8:key
      a,"":next
20    color4,1:color0,2,4:scnclr:vol8:di
      mv$(200)
30    char,12,1,"SPL[SPACE]ZNDR[SPACE]KL
      NKRS":print
40    print"[CRSR-DOWN][SPACE]De[SPACE]b
      edoeling[SPACE]van[SPACE]dit[SPACE
      ]spel[SPACE]is[SPACE]om[SPACE]het"
50    print"[SPACE]woord[SPACE]dat[SPACE
      ]de[SPACE]computer[SPACE]geeft[SPA
      CE]te[SPACE]raden."
60    print"[2xCRSR-DOWN][SPACE]In[SPACE
      ]de[SPACE]woorden[SPACE]ontbreken[
      SPACE]alle[SPACE]klinkers."
70    print"[2xCRSR-DOWN][SPACE]Het[SPAC
      E]is[SPACE]moeilijker[SPACE]dan[SP
      ACE]je[SPACE]denkt.[CRSR-DOWN]"
80    print"[2xCRSR-DOWN][SPACE]Je[SPACE
      ]kunt[SPACE]het[SPACE]spel[SPACE]z
      elf[SPACE]uitbreiden[SPACE]door[3x
      SPACE]meer[SPACE]woorden[SPACE]in[
      SPACE]de[SPACE]dataregels[SPACE]te
      [SPACE]"
90    print"[SPACE]zetten.":print"[2xCRS
      R-DOWN][SPACE]Je[SPACE]moet[SPACE]
      dan[SPACE]wel[SPACE]zorgen[SPACE]d
      at[SPACE]het[SPACE]laatste[2xSPACE
      ]woord[SPACE]STOP[SPACE]is."
100   char,12,22,"[CTRL-9]druk[SPACE]op[
      SPACE]een[SPACE]toets"
110   getkeyg$:scnclr:printchr$(142)
120   t=t+1:readd$:ifd$="stop"then140
130   v$(t)=d$:goto120
140   char,8,10,"[CTRL-2][CTRL-9]M[22xSP
      ACE]N"
150   char,8,11,"[SPACE][CTRL-0][COM-A]C
      CCCCCCCCCCCCCCCCCCCC[COM-S][CTRL-9]
      [SPACE]"
160   char,8,12,"[SPACE][CTRL-0]B":char,
      30,12,"B[CTRL-9][SPACE]"
170   char,8,13,"[SPACE][CTRL-0][COM-Z]C
      CCCCCCCCCCCCCCCCCCCC[COM-X][CTRL-9]
      [SPACE]"
180   char,8,14,"N[22xSPACE]M[CTRL-1][CT
      RL 0]"
190   w$="":ww$="":a=0:d=int(rnd(1)*(t-1
      )+1)
200   w$=v$(d):l=len(w$)
210   a=a+1:ifa>lthen260
220   restore340:forb=1to5:readk$
230   z$=mid$(w$,a,1):ifz$=k$thenz$="*":
      goto250
240   nextb
250   ww$=ww$+z$:goto210
260   char,((40-1)/2),12,ww$
270   char,9,15,chr$(27)+"t"
280   print"[SHIFT-CLR]":inputq$:ifq$<>w
      $thensound1,300,5:goto280
290   sound1,800,5:char,0,16,chr$(27)+"q
```

```
           "
300     char,10,12,"[20xSPACE]":goto190
310     datavlooientheater,koeienstal,auto
        wasserij,fietsband,tafelpoot,verke
        ersschool
320     datauitrit,trottoir,biologie,kippe
        ei,praatpaal,rijopleiding,computer
        ,stoel
330     dataradiouitzending,spaarvarken,ca
        ssettebandje,telefoon,stop
340     dataa,e,i,o,u:rem laatste regel!!!
        !!
```

** EINDE LISTING klinkers

REGEL 10	193	REGEL 180	2
REGEL 20	19	REGEL 190	75
REGEL 30	145	REGEL 200	3
REGEL 40	147	REGEL 210	81
REGEL 50	213	REGEL 220	12
REGEL 60	20	REGEL 230	147
REGEL 70	254	REGEL 240	196
REGEL 80	145	REGEL 250	212
REGEL 90	85	REGEL 260	117
REGEL 100	189	REGEL 270	198
REGEL 110	169	REGEL 280	236
REGEL 120	3	REGEL 290	37
REGEL 130	143	REGEL 300	201
REGEL 140	243	REGEL 310	133
REGEL 150	128	REGEL 320	124
REGEL 160	19	REGEL 330	131
REGEL 170	142	REGEL 340	145

Regenbui c16

Jos van Oijen heeft waarschijnlijk de afgelopen zomer veel naar buiten gekeken en inspiratie opgedaan voor het volgende spel. Na het opstarten van het spel wordt er een uitleg gegeven.

```
10      color4,1:color0,1:color1,2:vol8:pr
        intchr$(27)+"m"chr$(142)chr$(8):di
        mp(20):scnclr
20      color1,6,7:print"[CTRL-9][6xSPACE]
        [COM-*][CRSR-RIGHT][7xSPACE][CRSR-
        RIGHT][SHIFT-£][5xSPACE][COM-*][CR
        SR-RIGHT][7xSPACE][CRSR-RIGHT][SPA
        CE][COM-*][4xCRSR-RIGHT][SPACE]"
30      color1,6,6:print"[CTRL-9][SPACE][C
        TRL 0][SHIFT-£][3xSPACE][COM-*][CT
        RL 9][SPACE][CRSR-RIGHT][SPACE][CT
        RL 0][SHIFT-£][5xSPACE][CTRL-9][SH
        IFT £][SPACE][CTRL-0][SHIFT-£][3xS
        PACE][COM-*][CTRL-9][SPACE][CRSR-R
        IGHT][SPACE][CTRL-0][SHIFT-£][6xSP
        ACE][CTRL-9][2xSPACE][COM-*][3xCRS
        R-RIGHT][SPACE]"
40      color1,6,5:print"[CTRL-9][SPACE][5
        xCRSR-RIGHT][SPACE][CRSR-RIGHT][SP
        ACE][6xCRSR-RIGHT][SPACE][CTRL-0][
        SHIFT-£][5xSPACE][CTRL-9][SPACE][C
        RSR-RIGHT][SPACE][7xCRSR-RIGHT][SP
        ACE][CTRL-0][COM-*][CTRL-9][SPACE]
        [COM-*][2xCRSR-RIGHT][SPACE]"
50      color1,6,4:print"[CTRL-9][SPACE][C
        OM *][3xCRSR-RIGHT][SHIFT-£][SPACE
        ][CRSR-RIGHT][SPACE][COM-*][5xCRSR
        -RIGHT][SPACE][8xCRSR-RIGHT][SPACE
        ][COM-*][6xCRSR-RIGHT][SPACE][CRSR
        -RIGHT][CTRL-0][COM-*][CTRL-9][SPA
        CE][2xCRSR-RIGHT][SPACE]"
60      color1,6,3:print"[CTRL-9][6xSPACE]
        [CTRL-0][SHIFT-£][SPACE][CTRL-9][5
        xSPACE][2xCRSR-RIGHT][SPACE][3xCRS
        R-RIGHT][3xSPACE][COM-*][CRSR-RIGH
        T][5xSPACE][3xCRSR-RIGHT][SPACE][2
        xCRSR-RIGHT][SPACE][2xCRSR-RIGHT][
        SPACE]"
70      color1,6,2:print"[CTRL-9][SPACE][C
        TRL 0][SHIFT-£][SPACE][COM-*][CTRL
         9][SPACE][COM-*][2xCRSR-RIGHT][SP
        ACE][CTRL-0][SHIFT-£][5xSPACE][CTR
        L 9][SPACE][5xCRSR-RIGHT][CTRL-0][
        COM-*][CTRL-9][SPACE][CRSR-RIGHT][
        SPACE][CTRL-0][SHIFT-£][6xSPACE][C
        TRL 9][SPACE][2xCRSR-RIGHT][SPACE]
        [2xCRSR-RIGHT][SPACE]"
80      color1,6,3:print"[CTRL-9][SPACE][3
        xCRSR-RIGHT][CTRL-0][COM-*][CTRL-9
        ][SPACE][COM-*][CRSR-RIGHT][SPACE]
        [6xCRSR-RIGHT][SPACE][6xCRSR-RIGHT
        ][SPACE][CRSR-RIGHT][SPACE][7xCRSR
        -RIGHT][SPACE][2xCRSR-RIGHT][SPACE
        ][COM-*][CRSR-RIGHT][SPACE]"
90      color1,6,4:print"[CTRL-9][SPACE][4
        xCRSR-RIGHT][CTRL-0][COM-*][CTRL-9
        ][SPACE][CRSR-RIGHT][SPACE][6xCRSR
        -RIGHT][SPACE][COM-*][5xCRSR-RIGHT
        ][SPACE][CRSR-RIGHT][SPACE][7xCRSR
        -RIGHT][SPACE][2xCRSR-RIGHT][CTRL-
        0][COM-*][CTRL-9][SPACE][COM-*][SP
        ACE]"
100     color1,6,5:print"[CTRL-9][SPACE][5
        xCRSR-RIGHT][SPACE][CRSR-RIGHT][SP
        ACE][COM-*][5xCRSR-RIGHT][CTRL-0][
        COM-*][CTRL-9][SPACE][COM-*][3xCRS
        R-RIGHT][SHIFT-£][SPACE][CRSR-RIGH
        T][SPACE][COM-*][6xCRSR-RIGHT][SPA
        CE][3xCRSR-RIGHT][CTRL-0][COM-*][C
        TRL 9][2xSPACE]"
110     color1,6,6:print"[CTRL-9][SPACE][5
        xCRSR-RIGHT][SPACE][CRSR-RIGHT][7x
        SPACE][CRSR-RIGHT][CTRL-0][COM-*][
        CTRL-9][5xSPACE][CTRL-0][SHIFT-£][
        SPACE][CTRL-9][7xSPACE][CRSR-RIGHT
        ][SPACE][4xCRSR-RIGHT][CTRL-0][COM
         *][CTRL-9][SPACE]"
120     poke2023,12:print:print
130     color1,7,0:print"[CTRL-9][4xSPACE]
        [COM-*][2xCRSR-RIGHT][SPACE][2xCRS
        R-RIGHT][SPACE][2xCRSR-RIGHT][CTRL
         0][COM-*][CTRL-9][SPACE][CTRL-0][
        SHIFT-£]"
140     color1,7,1:print"[CTRL-9][SPACE][C
        TRL 0][SHIFT-£][2xSPACE][COM-*][CT
        RL 9][COM-*][CRSR-RIGHT][SPACE][2x
        CRSR-RIGHT][SPACE][3xCRSR-RIGHT][S
        PACE]"
150     color1,7,2:print"[CTRL-9][SPACE][4
        xCRSR-RIGHT][SPACE][CRSR-RIGHT][SP
        ACE][2xCRSR-RIGHT][SPACE][3xCRSR-R
        IGHT][SPACE]"
160     color1,7,3:print"[CTRL-9][SPACE][C
        OM *][2xCRSR-RIGHT][SHIFT-£][CTRL-
        0][SHIFT-£][CRSR-RIGHT][CTRL-9][SP
        ACE][2xCRSR-RIGHT][SPACE][3xCRSR-R
        IGHT][SPACE]"
170     color1,7,4:print"[CTRL-9][5xSPACE]
        [2xCRSR-RIGHT][SPACE][2xCRSR-RIGHT
        ][SPACE][3xCRSR-RIGHT][SPACE]"
180     color1,7,5:print"[CTRL-9][SPACE][C
        TRL 0][SHIFT-£][2xSPACE][COM-*][CT
```

```
     RL 9][COM-*][CRSR-RIGHT][SPACE][2x
     CRSR-RIGHT][SPACE][3xCRSR-RIGHT][S
     PACE]"
190  color1,7,6:print"[CTRL-9][SPACE][4
     xCRSR-RIGHT][SPACE][CRSR-RIGHT][SP
     ACE][2xCRSR-RIGHT][SPACE][3xCRSR-R
     IGHT][SPACE]"
200  color1,7,7:print"[CTRL-9][SPACE][C
     OM *][2xCRSR-RIGHT][SHIFT-£][CTRL-
     0][SHIFT-£][SPACE][CTRL-9][SPACE][
     COM-*][SHIFT-£][SPACE][3xCRSR-RIGH
     T][SPACE]"
210  print"[CTRL-9][4xSPACE][CTRL-0][SH
     IFT £][2xCRSR-RIGHT][COM-*][CTRL-9
     ][2xSPACE][CTRL-0][SHIFT-£][2xCRSR
     -RIGHT][CTRL-9][SHIFT-£][SPACE][CO
     M *]":poke2023,0
220  k=k+1:ifk=8thenk=0
230  color1,8,k:char,0,14,"(c)[2xSPACE]
     jos":char,29,16,"zomer[SPACE]1987"
240  char,5,16,"van":char,4,18,"oijen"
250  char,10,23,"joystick[SPACE]of[SP
     ACE]keyboard?"
260  getg$:ifg$<>"j"andg$<>"k"then220
270  ifg$="k"then300
280  sound1,800,3:char,8,24,"[CTRL-2]st
     op[SPACE]joystick[SPACE]in[SPACE]p
     oort[SPACE]1"
290  l$="d":r$="6":u$="5":d$="r":fora=1
     to1000:next:goto310
300  l$="[CRSR-LEFT]":r$="[CRSR-RIGHT]"
     :u$="[CRSR-UP]":d$="[CRSR-DOWN]"
310  scnclr:printchr$(14):print:print:p
     rint
320  print"[CTRL-2]Het[SPACE]is[SPACE]z
     omer[SPACE]1987[SPACE]en[SPACE]het
     [SPACE]regent[SPACE]dat[SPACE]het[
     SPACE]giet.":print
330  print"Jij[SPACE](het[SPACE]bolletj
     e)[SPACE]moet[SPACE]er[SPACE]voor[
     SPACE]zorgen[2xSPACE]dat[SPACE]de[
     SPACE]regen[SPACE]niet[SPACE]op[SP
     ACE]de[SPACE]grond[SPACE]komt."
340  print:print"Wanneer[SPACE]je[SPACE
     ]100[SPACE]druppels[SPACE]hebt[SPA
     CE]opgevangen,krijg[SPACE]een[SPAC
     E]ander[SPACE]level.":print
350  print"Als[SPACE]je[SPACE]dat[SPACE
     ]ook[SPACE]nog[SPACE]binnen[SPACE]
     twee[SPACE]minuten[2xSPACE]doet,[S
     PACE]krijg[SPACE]je[SPACE]50[SPACE
     ]bonus[SPACE]punten."
360  print:print"Wanneer[SPACE]er[SPACE
     ]20[SPACE]druppels[SPACE]op[SPACE]
     de[SPACE]grond[SPACE]zijn[SPACE]ge
     vallen,[SPACE]ben[SPACE]je[SPACE]a
     f."
370  char,12,20,"Druk[SPACE]op[SPACE]ee
     n[SPACE]toets":getkeyg$:scnclr:pri
     ntchr$(142)
380  scnclr:sc=0:m=0:l=0
390  k=0:l=l+1:ifl=9then930
400  k=k+1:ifk=8thenk=0
410  color1,3,k:char,16,12,"level"+str$
     (l):sound1,300+k*100,3
420  getg$:ifg$<>"[SPACE]"andg$<>"t"the
     n400
430  z=3571:c=z:s=0:t$=""
440  v$="[CTRL-9][SPACE][CRSR-DOWN][CRS
     R-LEFT][SPACE][CRSR-DOWN][CRSR-LEF
     T][SPACE][CRSR-DOWN][CRSR-LEFT][SP
     ACE][CRSR-DOWN][CRSR-LEFT][SPACE]"
450  b$="[CTRL-9][40xSPACE]":color0,7,4
     :sound1,300,45
460  scnclr:char,0,0,"[CTRL-1][CTRL-9][
     SPACE]tijd:[SPACE]0:00[2xSPACE]sco
     re:[5xSPACE]hi:[5xSPACE]mis:[4xSPA
     CE]":char,27,0,str$(hi)
470  char,19,0,str$(sc):char,36,0,str$(
     m)
480  char,0,1,b$:fora=1to23:char,0,a,"[
     SPACE]":char,39,a,"[SPACE]":next:c
     har,0,24,"[COM-7]"+b$+"[CTRL-1]"
490  restore:fora=1tol+1:readh,v:char,h
     ,v,v$:next
500  pokez-1024,113:pokez,81
510  fora=1to20:gosub760:p(a)=x:next:ti
     $="000000"
520  a=0
530  a=a+1:ifpeek(p(a))<>66thengosub760
     :p(a)=x
540  p(a)=p(a)+40
550  char,7,0,mid$(ti$,4,1):char,9,0,ri
     ght$(ti$,2)
560  ifpeek(p(a))=81thengosub710
570  ifpeek(p(a))<>160then600
580  ifp(a)<=3989then610
590  sound1,300,3:pokep(a)-40,32:m=m+1:
     char,36,0,str$(m):ifm=20then770:el
     se610
600  ifp(a)<4072thenpokep(a),66:pokep(a
     )-40,32:goto620
610  gosub760:p(a)=x:pokez,81
620  getg$:ifg$=""then700
630  ifg$=l$thenz=z-1:goto670
640  ifg$=r$thenz=z+1:goto670
650  ifg$=u$thenz=z-40:goto670
660  ifg$=d$thenz=z+40:else700
670  ifpeek(z)=160thenz=c
680  ifpeek(z)=66thengosub710
690  pokez-1024,113:pokez,81:ifz<>cthen
     pokec,32:pokec-1024,0:c=z
700  ifa<20then530:else520
710  sound3,1000,5:pokep(a)-40,32:s=s+1
     :sc=sc+1:char,19,0,str$(sc)
720  ifsc>hithenhi=sc:char,27,0,str$(hi
     )
730  ifs=100thent$=ti$:goto850
740  return
750  gosub760:p(a)=x:return
760  x=int(rnd(1)*304)+3193:ifpeek(x)<>
     32then760:elsereturn
770  char,7,12,"[CTRL-0][CTRL-2]jammer,
     [SPACE]je[SPACE]hebt[SPACE]verlore
     n!!":fora=1020to0step-10:sound1,a,
     1:next
780  char,7,14,"wil[SPACE]je[SPACE]nog[
     SPACE]een[SPACE]keer[SPACE](j/n)?"
790  getkeyg$:ifg$<>"j"andg$<>"n"then79
     0
800  ifg$="n"thenend:elsegosub810:goto3
     80
810  char,12,16,"druk[SPACE]op[SPACE]e
     en[SPACE]toets":poke239,0:getkeyg
     $
820  fora=0to13:char,0,13,chr$(27)+"d":
     char,0,13,chr$(27)+"d":sound1,460+
     40*a,3
830  print"[HOME]"chr$(27)+"i":color0,2
     ,4-(a/4):forb=1to50:nextb,a
840  fora=1to50:next:color0,1:return
850  ift$>"000200"then890
```

```
860     char,16,12,"[CTRL-8][CTRL-0]50[SP
        ACE]bonus[CTRL-9][CTRL-1]":sc=sc+
        50:char,19,0,str$(sc):ifsc>hithenh
        i=sc
870     char,27,0,str$(hi)
880     sound1,800,60:fora=1to10:sound2,60
        0+40*a,5:forb=1to50:nextb,a
890     char,2,12,"[CTRL-0][CTRL-2]profici
        at,[SPACE]je[SPACE]hebt[SPACE]leve
        l"+str$(l)+"[SPACE]gehaald!"
900     forb=30to400step20:fora=600to1000s
        tepb
910     sound1,a,3:sound2,a-110,2:nexta,b:
        gosub810:goto390
920     data10,18,29,18,19,13,19,6,24,10,1
        4,10,5,16,34,16,19,19
930     fora=50to150step10:forb=600to1000s
        tepa:sound1,b,3:sound2,b-110,2:nex
        tb,a
940     char,13,12,"gefeliciteerd!"
950     char,6,14,"je[SPACE]hebt[SPACE]all
        e[SPACE]levels[SPACE]gespeeld":for
        a=1to2000:next
960     char,2,12,"omdat[SPACE]ik[SPACE]ni
        et[SPACE]tegen[SPACE]mijn[SPACE]ve
        rlies[SPACE]kan"
970     char,3,14,"zal[SPACE]ik[SPACE]het[
        SPACE]programma[SPACE]vernietigen!
        ":fora=1to2000:next:poke65287,33
```

** EINDE LISTING regenbui

Missers

In de programma's Adresboek 64 en Diskbase 64 zijn enige karakters met printen verkeerd overgekomen:
-- Als er een ' staat moet u hiervoor typen : @
-- Als er een | staat moet u hiervoor typen : £

Het programma Disk-base is niet met een ingeschakelde cartridge in te typen.

In het programma Cardes c 16 zijn door hernummeren een paar foutjes geslopen. Hier moet u veranderen:

° In regel 1070 moet u veranderen run 3020 in : run 1080

° In regel 1450 moet u veranderen run 4060 in : run 1470

° Een aantal regelnummers moeten veranderd worden:

regelnr 1490 moet worden regelnr 5128
regelnr 1500 moet worden regelnr 5200
regelnr 1510 moet worden regelnr 5201
regelnr 1520 moet worden regelnr 5202
regelnr 1530 moet worden regelnr 5203

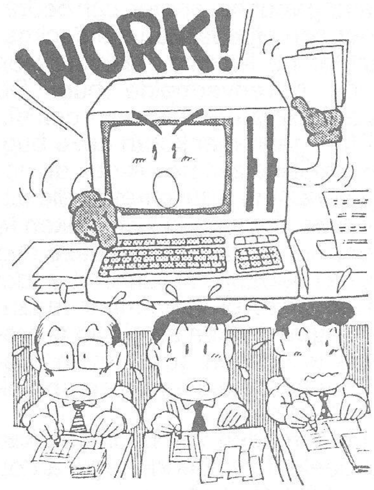
Niet vergeten na het veranderen de oude regelnummers er uit te halen!!

In het programma Tape-disk is ook door het hernummeren een fout onstaan. Deze fout is makkelijk te herstellen door in regelnummer 260 van het programma het getal 230 te wijzigen in 310 en het getal 240 in 320.

Wij zijn door de makers van de programma's op deze fouten gewezen, waarvoor onze dank.

In deze twintigste aflevering van onze populaire cursus Basic voor beginners komen we tot de ontdekking dat het ook voor de gevorderde programmeur nog steeds een uitdaging kan zijn om met een paar heel eenvoudige, maar slim bedachte, Basic regels een aardig programma op poten te zetten. In dit artikel vervolgt Jan Bodzinga zijn verhandeling over de Basic 'tekstverwerking', ofwel het programmeren met strings. De aflevering gaat vergezeld van een complete listing voor een aardig woordspel.

Basis Basic



Deel 20 Strings (II)

In de vorige les hebben we uitvoerig stilgestaan bij de beginselen van een Basic-string. Dit type variabele gedraagt zich nogal vreemd ten opzichte van de andere variabelen die we kennen in Basic en ze zijn daarom zonder meer de moeite waard. We gaan nu verder met de overige commando's die we in Basic op deze alfanumerieke variabelen los kunnen laten en we besluiten deze aflevering met een compleet Basic-programma waaraan iedereen, ook de programmeur zelf, heel wat plezier zal kunnen beleven.

Zoals bekend vraagt het werken met niet-rekenkundige data in Basic de nodige aanpassing. Het optellen van de getallen A en B waarbij respectievelijk voor A en B de waarden 45 en 623.4 in de computer aanwezig zijn, vormt meestal geen probleem. Hoewel we bij het calculeren met integers op wat afrondingen zullen stuiten, zorgt de 8-bits floating-point accumulator in RAM er wel voor dat we op z'n hoogst te kampen krijgen met een kleine onnauwkeurigheisfactor in dergelijke optellingen.
Het sommeren van alfanumerieke gegevens, bijvoorbeeld A$ = B$ + C$, waarbij B$ = "Een aardig " en C$ als waarde "meisje" heeft, resulteert in een string die qua optelling alleen de lengte van beide variabelen heeft samengevoegd, waardoor nu in A$ te lezen zal zijn "Een aardig meisje". Voor het overige zal er in deze expressie weinig kunnen worden opgeteld.
De primaire mogelijkheden met strings hebben we inmiddels wel onder de knie. Opdrachten zoals LEN, ASC() en CHR$() moeten als het goed is voor ons nu nog weinig problemen geven, terwijl daarmee ook het rangschikken ofwel sorteren van strings op hun ASCII waarde bekend zal zijn. Voorzover dit nog niet het geval is, raad ik iedereen aan de vorige lessen nog eens na te lezen, vooral de artikelen waarin het sorteren van met name strings, uitvoerig is beschreven.
Het is nu eenmaal zo in dit werk, dat er opeens een vlam van verlichting door je hersenen kan slaan op het moment dat je iets onbegrijpelijks ineens begrijpt. Daarom kan ik er niet voldoende op hameren om ook over de meest simpele commando's in Basic beslist niet zomaar heen te stappen, maar de dingen werkelijk te lezen, te proberen op de computer en eventueel nogmaals te lezen, te wijzigen, te runnen en steeds maar weer opnieuw te herhalen tot de werking volledig duidelijk is.
Uiteindelijk is dat de beste manier om het programmeren helemaal onder de knie te krijgen.

Functies

Er zijn nog een paar functies mogelijk met strings, die we tot op heden hebben gelaten voor wat ze waren. Zo nu en dan konden we er niet onderuit en moesten we één of meer LEFT$(), MID$() en RIGHT$() opdrachten in een programma verwerken. Maar in principe hebben we daar nog geen aandacht aan besteed. We zullen dat in deze les goedmaken.
Daarmee wordt het hele hoofdstuk strings afgesloten en kunnen we ons beschouwen als nagenoeg volleerde Basic-programmeurs. Voordat we daarmee beginnen eerst even een kleine zijsprong naar de functie STR$(). Deze functie voert het tegengestelde uit van de functie VAL(), waarbij de numerieke waarde uit een string wordt getrokken. Met STR$() wordt vanuit een floating-point variabele een waarde geconverteerd naar een string. Als je het goed beschouwt een onzinnige functie, maar aan de andere kant zeker handig en met de kwaliteti van het programmeerwerk vandaag de dag bijna een onontbeerlijke functie. We kunnen in ieder geval goed overweg met het printen van getallen als we deze STR$() functie beheersen.

STR$()

Zoals gezegd is deze stringfunctie het tegengestelde van de functie VAL$(), waarmee de waarde van een string naar een numerieke variabele kan worden overgebracht.
Nadat dit is gebeurd kan er natuurlijk niet meer met de waarden worden gerekend. Ze zijn vanaf dat moment geheel te beschouwen als string, ofwel tekstvariabele.
Met de functie STR$() worden getallen of numerieke expressies overgebracht naar een string. De syntax van deze functie wijkt niet af van andere functies en is:

STR$(num.expressie)

De wiskundige uitdrukking tussen de haken kan elke vorm aannemen die geldig is bij een floating-point variabele. Over de FLP zullen we het een volgende keer hebben. De waarde die daarmee wordt bedoeld kan maximaal +/- 1.7 E38 zijn en geloof me, dat is een heleboel. Jammer is het alleen, dat STR$() hetzelfde buffer gebruikt als PRINT, waardoor een niet al te grote nauwkeurigheid wordt bereikt tijdens de conversie. Het gebeurt nogal snel, dat een uitdrukking wordt omgezet in een zogenoemde 'wiskundige

notatie' en daar heb je als huis-tuin-en-keuken-programmeur weinig mee te maken.
De expressie **A$ = STR$(0.005)** wordt daardoor helaas niet omgezet in **A$ = "0.005"**, maar in een beetje algebra, ofwel **A$ = "5E-03"**.
STR$() kan goed worden gebruikt voor het maken van nette, afgeronde print-string voor het printen van bedragen met een afgepast aantal decimalen achter de komma. Maar kijk uit voor de bovenvermelde bug. De meeste Basic-programma's, - ook die van IBM - gaan mank aan deze bug. En om eerlijk te zijn heb ik ook de nodige programma's geschreven die uiteindelijk de E38 bug in zich bleken te hebben. Het aardige is, dat deze bug er pas na heel lange tijd uitkomt, zodat je als argeloze programmeur maandenlang kunt denken een feilloos programma te hebben geschreven. Gelukkig heeft Commodore hieraan ook een beetje schuld.
Voor het afronden van bedragen kan bijvoorbeeld de volgende regel aardige diensten bewijzen:
20 PRINT STR$(145)+".00"
Ook het voor-nullen van getallen kan alleen gebeuren met hulp van STR$. Het is een aardige oefening om zelf een routine te proberen die alle getallen voorziet van twee decimalen, waardoor deze getallen als bedragen, netjes onder elkaar in kolommen geprint kunnen worden.

LEFT$()

De functie LEFT$ haalt een gedeelte uit een string en zet dit in een andere string. Hoewel op het eerste gezicht misschien niet erg spectaculair is dit één van de functies waarbij het werkelijk fantastisch is wat de Commodore voor ons kan doen. LEFT$() haalt het linkergedeelte uit een string. Vergelijkbare functies zullen we straks bekijken. Dit zijn MID$() en RIGHT$(). De principes van deze drie functies zijn identiek. Deze functies worden veel gebruikt bij het manipuleren van tekst en alfanumerieke data in Basicprogramma's.
De officiële syntax van LEFT$() is:
LEFT$(stringexpressie, numerieke expressie)
Beide uitdrukkingen moeten natuurlijk geldig zijn en door Basic worden geaccepteerd. De stringexpressie moet uiteindelijk resulteren in een 'echte' string, hoewel bijv. ook **STR$()** in deze functie kan worden gebruikt. De lengte van de string kan bij Commodore beslist niet langer zijn dan 256 bytes. Maar veel strings zullen deze lengte nooit halen, dus daar hoeven we ons geen zorgen over te maken.
De maximale waarde van de numerieke expressie kan daarom ook niet hoger liggen dan 256 bytes. De functie van deze numerieke expressie is, dat daarmee wordt aangegeven hoeveel bytes van de stringexpressie vanaf de linker(LEFT)kant in de nieuwe string moeten worden gezet.
Het beste wordt deze functie geïllustreerd met een paar voorbeelden, die samen alle mogelijkheden van LEFT$() laten zien:
A$ = LEFT$("Sandra is een leuk kind" ,7) geeft voor A$ "Sandra"; uiteraard kan het volgende ook:
B$ = "Sandra is een leuk meisje": A$ = LEFT$(B$,7), waarbij hetzelfde resultaat wordt bereikt voor A$.
Andere mogelijkheden zijn:
PRINT LEFT$("HALLO"+"HARRY",3) zet "HAL" op het scherm, terwijl **PRINT LEFT$("HALLO"+"HARRY",7)** op het scherm "HALLOHA" zal tonen. **PRINT LEFT$("HALLO",70)** laat gewoon "HALLO" zien en **LEFT$("HALLO",0)** geeft een lege string met nul bytes.
Ook andere mogelijkheden zijn te benutten met LEFT$, kijk maar eens naar de volgende voorbeelden:

```
10 PRINT LEFT$(X$+"
     ",10); :REM TAB-simulatie
20 PRINT LEFT$(STR$(X+5),12);:
     REM conversie vanuit
     numeriek
30 PRINT L;: PRINT LEFT$("      ",
     10- LEN(STR$(L)));:REM TAB
```

Vooral de laatste mogelijkheid herbergt een verassing die niet direct in het oog springt. Maar met dit statement zijn we in staat van een reeks getallen nette kolommen te maken op het scherm. Ze zijn dan weliswaar links uitgelijnd, maar evengoed een snelle manier om een tabel te maken. De functie is bijzonder goed toe te passen bij het gebruik van printroutines of het formateren van strings op het scherm. Ook andere manipulaties met strings kunnen met hulp van LEFT$ gemakkelijk tot stand komen.

RIGHT$()

De RIGHT$ functie heeft in principe dezelfde kenmerken en mogelijkheden als LEFT$. Het enige verschil met de vorige functie is de kant van de string van waaraf het gedeelte wordt genomen. RIGHT$ nemt het meest rechter gedeelte van de string, en gezien de naam van de functie zullen we ook niet veel anders hebben verwacht. De syntax van RIGHT$ is gelijk aan die van LEFT$, waarbij uiteraard ook dezelfde limieten gelden met betrekking tot de uiterste waarden.
Hoewel bijna gelijk aan LEFT$ hebben we toch wat aparte voorbeelden voor RIGHT$ kunnen bedenken:

10 A$ = RIGHT$("ABCDEFGHIJ",4) geeft A$="GHIJ"
20 PRINT RIGHT$("ALEXANDER",5) wordt "ANDER"
30 FOR I = 1 TO LEN(W$) : S$=S$+"-": NEXT :REM oplengen met ----
40 S$= LEFT$(S$,J-1)+L$+RIGHT$(S$,LEN(S$)-J):REM zie verder

In het verloop van deze les zullen we de praktische voorbeelden van zowel LEFT$ als RIGHT$ nog wel tegen komen. Grappige functies met een forse impact.

MID$()

De laatste functie die we in dit stringverband willen bespreken is een beetje ingewikkelder dan LEFT$() en RIGHT$(). Het gaat om de functie MID$(), die in feite dezelfde werking heeft als de vorige twee functies, maar in staat is om een substring te formeren uit het midden van een bestaande string. We zullen daarvoor ook wat meer parameters aan deze functie moeten meegeven.
De syntax voor MID$() luidt als volgt:

MID$(stringexpressie, numerieke expressie, [numerieke expressie])

Een hele mond vol, maar in de praktijk valt het allemaal wel mee. Het is logisch, dat we bij een dergelijke functie minimaal 2 parameters moeten gebruiken: als eerste om aan te geven welke string er als uitgangspunt moet worden gebruikt, en een tweede (numerieke) parameter om de interpreter te vertellen op welk punt in de string we de functie willen laten beginnen. De derde, ook numerieke parameter dient om aan te geven hoeveel bytes van de oorspronkelijke string moeten worden overgebracht naar de nieuw te maken string. De haken om deze parameter geven aan, dat het niet per sé noodzakelijk is, om deze derde parameter ook te gebruiken in de functie. Als hij wordt weggelaten zal MID$() vanaf het punt wat door de tweede parameter wordt aangegeven tot aan het einde van de originele string aan de nieuw te maken string toewijzen. In het laatste geval blijkt de werking dus identiek aan RIGHT$(), die daardoor bijna overbodig wordt.
MID$() kan aan de hand van een paar aardige voorbeelden het beste worden geïllustreerd:

10 A$ = MID$("1234567890",2,3) :REM wordt A$="234"

20 B$ = "1234567890":A$= MID$(B$,4,2): REM wordt A$="45"
30 B$ = "1234567890":A$=MID$ (B$,6,12):REM wordt A$="67890"
40 B$ = "1234567890":A$=MID$ (B$,4):REM wordt A$="4567890"
50 B$ = STR$(1234):a$=MID$ (B$,2):REM verwijder spatie

Tot zover de normale voorbeelden van de functie MID$(). Let met name op regel 30, waaruit blijkt, dat ook als de derde parameter groter is dan het einde van de originele string er door MID$() slechts tot het laatste karakter van de string wordt doorgegaan.
Ook regel 50 geeft een aardige toepassing weer van MID$(). Het gebruik van de functie STR$() heeft namelijk de eigenschap om bij positieve getallen een extra spatie in de string te stoppen. Dit is geen bug, maar door Basic bedoeld voor het plusteken, dat voor het gemak wordt weggelaten. Bij negatieve getallen zal bij gebruik van de functie STR$() geen ruimte overblijven voor een spatie. Het voorbeeld in regel 50 geeft voor B$ dus de waarde " 1234", ofwel een string met totaal 5 bytes, terwijl het gaat over een getal van vier digits. Omdat we in A$ pertinent zonder spaties het gehele getal "1234" willen zien, gebruiken we de functie MID$(B$,2) zonder derde parameter en geven daarmee aan, dat het eerste byte van B$ kan komen te vervallen. We moeten er daarbij van uitgaan, dat het getal positief is, anders zullen we in de problemen komen. Een eenvoudige controle kan daarvoor zorgen, waardoor de regel wordt uitgebreid met een IF constructie:
50 B=1234:B$=STR$(B): A$=B$: IF B >0 THEN A$=MID$(B$,2)
Met deze handelwijze is ook het eventueel negatieve getal keurig opgevangen.

Voorbeeld

Een aardig en daarom handig toe te passen voorbeeld met gebruik van MID$() wil ik u niet onthouden. Het is een wijze van programmeren waar je als beginner waarschijnlijk niet zo snel opkomt, omdat het geheel een beetje buiten de standaardkanalen ligt. Het gaat om het uitzoeken van min of meer statische data, die normaal in een array zou worden gezet. We kunnen dat goed bekijken aan de hand van de 12 maanden van het jaar. Deze namen zijn statisch, dat wil zeggen ze veranderen meestal niet en we weten zelfs hoeveel het er zijn. Uitgaande van deze twee feiten kunnen we in Basic een array opbouwen met precies twaalf indices, waarin we van 0 tot 11, of -gemakkelijker- van 1 tot 12 de namen van de verschillende maande stoppen. Al met al standaard, ruimtevretend en uitvoerig programmeerwerk, maar zeker niet verkeerd. Een andere manier kunnen we goed toepassen met hulp van MID$(). We kiezen hier voor de methode waarbij we standaard uitgaan van de maandafkortingen, omdat dit duidelijker is.
We beginnen in het programma om een string te maken, waarin de maanden afgekort) in de goede volgorde achter elkaar worden gezet:
10 MD$="JanFebMaaAprMeiJunJulAugSepOktNovDec"

We hebben daarmee een string met een lengte van precies 12 x 3 is 36 bytes. De afkortingen zijn op zichzelf ook in het nederlands niet ongebruikelijk, alleen voor de maand maart is een afkorting van MAA een beetje zot. De hoofdletters hebben we voor de duidelijkheid ingevoegd, maar zijn uiteraard niet nodig.
Nu het voorbeeld:

```
10 MD$="JanFebMaaAprMeiJun
      JulAugSepOktNovDec"
20 INPUT "Geef maandnummer
      :";MN
30 IF MN < 1 OR MN > 12 THEN 20
      :REM controle
40 M$ = MID$(MD$,3*MN-2,3)
50 PRINT "MAAND : ";M$
```

Kort, maar krachtig, en dat alles met hulp van een simpele functie als MID$(). De echte truc schuilt natuurlijk in regel 40. In deze programmaregel wordt van de string MD$, via een rekenkundige expressie '3*MN-2' berekend welk gedeelte we van de totale string nodig hebben aan de hand van het maandnummer. Een simpele oplossing, en als je eenmaal de techniek onder de knie hebt ook één die zich er voor leent veel en vaak te worden gebruikt voor statische gegevens, die daarbij natuurlijk allemaal wel even lang moeten zijn. Voor het werken met complete namen is deze techniek ook zeker toe te passen, maar dan zal de string voor namen korter dan de langste maand (september) met spaties moeten worden opgelengd. Een leuke oefening om zelf zeker eens wat mee te expirimenteren.

Woordspel

Om met de strings ook in de praktijk wat aardigs te kunnen doen, hebben we speciaal voor deze aflevering een kort Basic-programma geschreven, dat de kern vormt voor een bekend spel met woorden. Normaal wordt dit spel gespeeld zonder computer, en moet aan de hand van een gekozen woord per letter het volledige woord worden geraden. Het feit dat we nu nogal intensief met strings bezig zijn geweest vormt een aanleiding om dit spel in simpele vorm ook voor de computer als Basic-programma zelf te gaan schrijven.
Aan de hand van de inmiddels beschikbare Basic-opdrachten hebben we een elementaire listing opgesteld, waarmee het wooord-spel, ook wel bekend als 'Galgje', tegen de computer kan worden gespeeld.
Het programma voor dit spel is opgebouwd uit diverse modules. Als eerste natuurlijk de initialisatie van variabelen. Daarna het inlezen van de database waarin de woorden zijn ondergebracht. Vervolgens een routine om 'willekeurig' een woord te kiezen uit alle beschikbare woorden, terwijl daarna dit woord wordt bewerkt en gereed gemaakt voor het raadspel. Dan wordt de invoer van een letter gevraagd en afgehandeld. Aan de hand van het al of niet goed/fout zijn van deze invoer wordt gevraagd het complete woord in te voeren. Deze input wordt getest en vergeleken met het originele woord. Is dit fout, dan begint het raden opnieuw met de routine voor gebruikers-input.
Als we dat op een rij zetten komen we tot de volgende onderscheiden delen:

WOORDSPEL

10 - 190 Initialisatie
200 - 240 Inlezen woorden
MAIN 300 - 350 Random zoeken woord
400 - 600 Gebruikersinput
SUBS 1000 - 1060 Zoek letter in woord
1100 - 1130 Tussenvoegen OK letter
1200 - 1240 Vergelijk woordinput
DATA 2000 - Databank met woorden.

We beginnen met het eerste gedeelte van het programma. De definitie van de verschillende variabelen en ook van de array W$(), waarin de woorden straks zullen worden ingelezen:

```
10   REM
     ************************************
20   REM Werken met strings
30   REM Woordspel / GALGJE
40   REM 1988 j/bodzinga C-Info
99   REM
     ************************************
100  REM initialisatie
110  DIM W$(200) :REM aantal
     woorden
120  I=0: REM teller
130  SS$="--------------------------------"
     :REM STREEP
140  BT=0:REM aantal beurten
150  OK=0:REM vlag goed woord
160  GD=0: REM VLAG OP JUISTE
     LETTER
```

Tot hier toe weinig nieuwe dingen. De declaratie van de verschillende variabelen, I, BT, OK en GD dienen meer om de programmeur achteraf bij wijzigingen enzovoort behulpzaam te zijn. Het enige dat hier echt nodig is, is de DIMensie van de array W$(), waarmee ruimte wordt gemaakt voor 201 woorden. (regel 110)
De string SS$, waarin allemaal streepjes zijn gezet, gaat straks functioneren om het te raden woord aan de spelers op het scherm te tonen. Voor ieder streepje staat dan een nog niet geraden letter.

DATA

Als tweede gedeelte nemen we de databank van ons spel onder de loep. Het is evenals de vorige module weinig spectaculair:

```
2000 DATA
     commodore,knokken,door,aan
     ,borstbeeld,groot
2010 DATA jodelen,taxi,mini,
     maximum,minimaal,rochelen
2020 DATA
     schaatsen,plukken,kruipen,juf
     frouw,beer
2030 DATA
     bovenkant,autobus,uitgeverij,
     meisje,computer
2040 DATA groenekool, krullebol,
     beddesprei, medelijden
3000 DATA -1
```

Het geheel is te beschrijven als een 'statische' databank, waardoor het geen noodzaak is om de woorden op te slaan in een apart bestand. Wel is het natuurlijk mogelijk de woorden uit de listing aan te vullen met eigen vondsten -de nederlandse taal kent per slot van rekening tienduizenden woorden- en zelfs heel persoonlijke gegevens zoals de naam van je vriendin kunnen erin worden verstopt. Als er maar rekening mee wordt gehouden, dat het totaal aantal woorden niet boven de 201 komt te liggen, want daarop is de array W$ gedimensioneerd. De data wordt afgesloten met een "-1" string, waarop bij het inlezen in de volgende module wordt getest:

```
200  REM inlezen woorden in array
210  READ W$(I)
220  IF W$(I) "-1" THEN I=I+1:GOTO
     210
230  PRINT CHR$(147): REM Clear
     screen
235  I=I-1: REM Teller juiste aantal
240  PRINT "Totaal aantal woorden
     :";I
```

Simpel en recht voor z'n raap worden in regel 210 en 220 alle beschikbare woorden sequentieel uit de DATA-regels gelezen. Daarna wordt de totaalteller (I) goed gezet en voor de aardigheid even op het scherm geprint. Het spel kan beginnen:

```
300  REM begin spel
310  REM zoeken woord
320  RW$ = "":GD=0: OK=0: BT=0
330  W$ = W$(RND(0)*I)
340  L = LEN(W$) : REM LENGTE
     WOORD
350  HW$=LEFT$(SS$,L) :REM
     raadwoord
```

Het begin van het echte spel maakt eerst een lege string in RW$. Daarna wordt met de RND() functie een woord gekozen uit de beschikbare hoeveelheid woorden (I). De variabele L krijgt de lengte van het woord en in regel 350 wordt HW$ gevuld met evenveel streepjes uit de string SS$ als het originele te raden woord lang is (L). Nu printen we het inputscherm:

```
400  REM woordscherm
410  PRINT CHR$(147) : BT=BT+1
420  PRINT:PRINT:PRINT
430  PRINT" RAAD het woord
     beurt :";BT
440  PRINT:PRINT "Woord: ";HW$
450  PRINT"Geraden letters : ";RW$
460  PRINT:PRINT:PRINT
500  PRINT"Welke letter : ";
510  INPUT L$
550  RW$=RW$+L$+" "
560  GOSUB 1000 : REM uitzoeken
     woord
570  IF GD=1 THEN GOSUB 1200
580  IF OK =1 THEN 300
590  IF BT > 10 THEN PRINT
     :PRINT"Teveel beurten /
     woord : ";W$:GOTO 300
600  GOTO 400
```

Uiteindelijk is dit het hoofdgedeelte van het programma. BT wordt eentje hoger en het scherm wordt schoon. Na wat print-opdrachten krijgen we de huidige stand van het woord HW$ op het scherm. Daaronder de vraag een letter (L$) in te toetsen. Als dit is gebeurd, wordt deze nieuwe letter (L$) direct in de string RW$ gezet, om bij de volgende beurt in regel 450 aan de speler te kunnen tonen welke letters inmiddels door hem geraden zijn. Er wordt in dit programma dus niet getest of een ingetypte letter al eerder is ingevoerd! De test van de ingevoerde letter wordt straks in subroutine 1000 vv. uitgevoerd:

```
1000 REM uitzoeken woord/letter
1010 GD=0: REM goed/fout vlag
     letter
1020 FOR TT=1 TO L : REM loop
     lengte woord
1030 IF L$ = MID$(W$,TT,1) THEN
     GOSUB 1100
1040 NEXT
1050 IF GD=1 THEN PRINT:PRINT
     HW$
1060 RETURN
```

De vlag voor een juiste of onjuiste letter (GD) wordt vooraf altijd op fout gezet (GD=0). Daarna wordt bekeken of de letter wellicht één of meer keren in het woord (W$) aanwezig is. Om dit te doen wordt in een lus TT met hulp van de functie MID$() telkens 1 letter uit W$ gehaald en vergeleken met de ingevoerde letter (L$). In de volgende subroutine wordt daarna bij een goede letter de streep op de juiste plaats in HW$ vervangen door de letter. Daarbij wordt ook vlag GD op 1 gezet. In regel 1050 zien we, dat bij het intypen van een juiste letter het opnieuw gemaakte woord HW$ nu met de juiste ingevoerde letter op het scherm wordt geprint. De subroutine waarin dit woord HW$ wordt samengesteld vinden we vanaf regel 1100:

```
1100 REM tussenvoegen letters in
     raadwoord
1110 GD =1
1120 HW$=LEFT$(HW$,TT-1)+L$+RI
     GHT$(HW$,L-TT )
1130 RETURN
```

Een eenvoudige routine, die nu eens gebruik maakt van de functies LEFT$ en RIGHT$. Helaas kent Commodore Basic geen INSTRING() functie waardoor deze subroutine bijna overbodig zou worden. We hebben bij het ingaan van deze routine als waarden beschikbaar: L voor de lengte van W$ en HW$, GD als vlag en TT als positie in de string waar een juiste letter (L$) voorkomt. Met deze feiten wordt in regel 1120 netjes de string HW$ in drieën gekapt en opnieuw geplakt met de juiste letter gepositioneerd. Met de vlag GD=1 komen we terug in subroutine 1000 vv en vandaar in het hoofdgedeelte:

```
560  GOSUB 1000 : REM uitzoeken
     woord
570  IF GD=1 THEN GOSUB 1200
580  IF OK =1 THEN 300
590  IF BT > 10 THEN PRINT
     :PRINT"Teveel beurten /
     woord : ";W$:GOTO 300
600  GOTO 400
```

Als GD=1 (regel 570) is er nu een goede letter ingetypt, en zullen we de speler moeten vragen of hij het woord misschien al weet. Dit gebeurt in subroutine 1200 en verder. Voordat we daarnaar kijken zullen we eerst het laatste stuk van het hoofdgedeelte af-

ronden. De OK vlag, die we in regel 580 tegenkomen geeft aan of het complete woord al dan niet geraden is. Is dit het geval (OK=1) dan gaan we door met regel 300 waar een nieuw woord wordt opgezocht.

Is het woord nog niet geraden (OK=0) dan testen we in regel 590 eerst of er inmiddels niet meer dan 10 beurten zijn verstreken, waarbij de computer de oplossing geeft en ook met regel 300 begint aan een nieuw woord. Voldoet geen van de vorige condities, dan kunnen we verder met regel 400, om opnieuw een keuze uit de beschikbare letters te laten maken door de spelers.
Nu nog routine 1200 waarin wordt getest of het complete woord kan worden geraden:

```
1200 REM vergelijk woord
1210 OK=0
1220 INPUT "Welk woord ";W1$
1230 W$ =W1$ THEN PRINT
     :PRINT"Goed geraden in
     ";BT;" beurten":OK=1
1240 RETURN
```

De woord-vlag (OK) wordt op nul gezet, waarna regel 1220 vraagt het gehele te raden woord in te voeren in W1$. Regel 1230 vergelijkt deze input met het originele woord uit de array (W$) en trekt al naar gelang de uitkomst conclusies. Als het woord goed is, dan volgt daarop een melding met het aantal beurten en wordt de OK vlag op 1 gezet. Bij het falen van de oplossing gebeurt er weinig in deze subroutine.

Daarmee zijn we aan het eind gekomen van dit programma en ook van deze aflevering. Ik hoop dat het werken met strings voor iedereen duidelijk is geworden en ik wil een ieder uitdagen op grond van bovenstaande listing een echt spel in elkaar te steken, compleet met alles tests, een keurige galg en een volledige database met woorden. Ook de schermopmaak is niet erg fraai in deze voorbeeldlisting, maar de principes lijken me duidelijk genoeg om mee aan de slag te kunnen. Sterkte en ik zie de listings met spanning tegemoet.
Jan Bodzinga

Complete listing :

```
10   REM
     ************************************
20   REM Werken met strings
30   REM Woordspel / GALGJE
40   REM 1988 j/bodzinga C-Info
99   REM
     ************************************
100  REM initialisatie
110  DIM W$(200) :REM aantal
     woorden
120  I=0: REM teller
130  SS$="--------------------------------"
     :REM STREEP
140  BT=0:REM aantal beurten
150  OK=0:REM vlag goed woord
160  GD=0: REM VLAG OP JUISTE
     LETTER
200  REM inlezen woorden in array
210  READ W$(I)
220  IF W$(I) "-1" THEN I=I+1:GOTO
     210
230  PRINT CHR$(147): REM Clear
     screen
235  I=I-1: REM Teller juiste aantal
240  PRINT "Totaal aantal woorden
     :";I
300  REM begin spel
310  REM zoeken woord
320  RW$ = "":GD=0: OK=0: BT=0
330  W$ = W$(RND(0)*I)
340  L = LEN(W$) : REM LENGTE
     WOORD
350  HW$=LEFT$(SS$,L) :REM
     raadwoord
400  REM woordscherm
410  PRINT CHR$(147) : BT=BT+1
420  PRINT:PRINT:PRINT
430  PRINT" RAAD het woord
     beurt :";BT
440  PRINT:PRINT "Woord: ";HW$
450  PRINT"Geraden letters : ";RW$
460  PRINT:PRINT:PRINT
500  PRINT"Welke letter : ";
510  INPUT L$
550  RW$=RW$+L$+" "
560  GOSUB 1000 : REM uitzoeken
     woord
570  IF GD=1 THEN GOSUB 1200
580  IF OK =1 THEN 300
590  IF BT > 10 THEN PRINT
     :PRINT"Teveel beurten /
     woord : ";W$:GOTO 300
600  GOTO 400
1000 REM uitzoeken woord/letter
1010 GD=0: REM goed/fout vlag
     letter
1020 FOR TT=1 TO L : REM loop
     lengte woord
1030 IF L$ = MID$(W$,TT,1) THEN
     GOSUB 1100
1040 NEXT
1050 IF GD=1 THEN PRINT:PRINT
     HW$
1060 RETURN
1100 REM tussenvoegen letters in
     raadwoord
1110 GD =1
1120 HW$=LEFT$(HW$,TT-1)+L$+RI
     GHT$(HW$,L-TT )
1130 RETURN
1200 REM vergelijk woord
1210 OK=0
1220 INPUT "Welk woord ";W1$
1230 W$ =W1$ THEN PRINT
     :PRINT"Goed geraden in
     ";BT;" beurten":OK=1
1240 RETURN
2000 DATA
     commodore,knokken,door,aan
     ,borstbeeld,groot
2010 DATA jodelen,taxi,mini,
     maximum,minimaal,rochelen
2020 DATA
     schaatsen,plukken,kruipen,juf
     frouw,beer
2030 DATA
     bovenkant,autobus,uitgeverij,
     meisje,computer
2040 DATA groenekool, krullebol,
     beddesprei, medelijden
3000 DATA -1
```

In het vorige nummer van Commodore INFO heeft Peter Heinckens ons laten kennis maken met een manier om sub-programma's te schrijven voor het invoeren van functies in programma's. Ditmaal gaat hij verder in op een tweede probleem: het omzetten van een formule van de infix- naar de postfix-notatie.

Omzetten van functie-notatie

Functie-analyse (2)

Het is noodzakelijk dat we ten volle begrijpen hoe een uitdrukking er uitziet, alvorens we ermee gaan 'goochelen'. Een goed middel hiertoe is de uitdrukking in de zogenaamde Backus-Naur-notatie te schrijven. Meer informatie over deze notatie vindt u in de appendix.

Laat ons eens kijken naar de Backus-Naur-formulering (BNF) van "expressie" (in de infix-vorm):

```
<expressie> == [<teken>] <term> {<plus operator> <term>}
<term>      == <factor> {<maal operator> <factor>}
<factor>    == (  <tekenloos getal> | <parameter> |
                ( <expressie> ) )
<plus operator> == ( + | - )
<maal operator> == ( * | / )
```

Met het deelteken bedoelen we 'div' als we met gehele getallen werken. In dit geval kunnen we ook 'mod' in onze definitie inlassen. Merk ook op dat er op dit moment nog geen functies toegelaten zijn in <expressie>.
Later zullen we deze definitie evenwel uitbreiden
Als we nu een bepaalde formule willen evalueren, moeten we enkel iedere stap in de BNF vertalen in een overeenkomstige procedure, en dan de geschikte aanroepen. Het volgende algoritme illustreert dit:

1. EVALUEER EXPRESSIE

(i) evalueer term

(ii) WHILE nog elementen DO
(i) evalueer term
(ii) evalueer operator (plus)

Bemerk dat in de tweede stap de 'term-evaluatie'-routine vóór de 'plus-operator'-routine aangeroepen wordt, terwijl deze laatste het eerst genoemd wordt in de BNF. Dit komt doordat we in de postfix-notatie de operator na zijn argumenten moeten plaatsen.

2. EVALUEER TERM

(i) evalueer factor

(ii) WHILE nog elementen DO
(i) evalueer factor
(ii) evalueer operator (maal)

3. EVALUEER FACTOR

(i) CASE next element OF

getal : evalueer getal
parameter : evalueer parameter
haakje : evalueer expressie
teken : laat teken inwerken op uitdrukking

Indien 'next-element' een teken is, dan tellen we de uitdrukking waarop het teken slaat bij nul op, of trekken ze ervan af, al naar gelang het teken respectievelijk een plus of een min is.

4. EVALUEER OPERATOR

(i) schuif operator op de operator-stack
(ii) laad de next-stack met "operator".

5. EVALUEER GETAL

(i) schuif getal op de number-stack
(ii) laad de next-stack met "number".

6. EVALUEER PARAMETER

(i) laad de next-stack met "parameter".

Bemerk dat we opnieuw gebruik gemaakt hebben van recursie. Dit is logisch, aangezien de BNF dit ook doet. Als we nu ook functies willen beschouwen, wordt het wel iets moeilijker. Vooreerst moet de BNF van 'expressie' uitgebreid worden:

```
<expressie> == [<teken>] <term> {<plus operator> <term>}
<term>      == <factor> {<maal operator> <factor>}
<factor>    == (  <tekenloos getal> | <parameter> |
               (<expressie>) | <functie> (<expressie>) )
<functie>   == ( inc | dec )
<plus operator> == ( + | - )
<maal operator> == ( * | / )
```

Natuurlijk kunnen er ook andere functies dan 'inc' en 'dec' gebruikt worden. In ons algoritme dienen we enkel de procedure FACTOR te wijzigen:

3. EVALUEER FACTOR

(i) CASE next element OF

getal : evalueer getal
parameter : evalueer parameter
haakje : evalueer expressie
functie : evalueer functie
teken : laat teken inwerken op uitdrukking

7. EVALUEER FUNCTIE

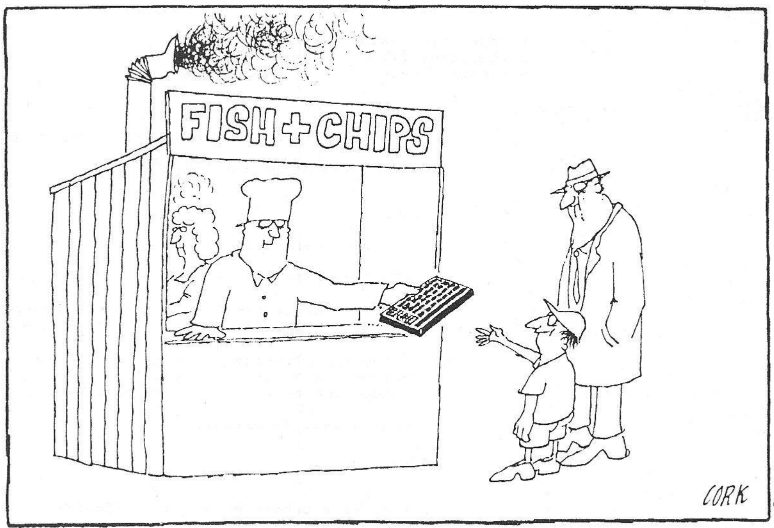
(i) Lees de functie in (f)

(ii) Evalueer expressie
(iii) Schuif f op de function-stack
(iv) schuif "function" op next-stack.

Het inlezen van de functies kan voor (nog meer) moeilijkheden zorgen: vooreerst moeten we in staat zijn het einde van de functienaam te bepalen. Dit kunnen we doen door te kijken of we een haakje bereikt hebben. Verder moeten we ook de functie herkennen. Hiertoe kunnen verschillende methodes gebruikt worden: we zouden gebruik kunnen maken van een CASE-instructie, maar dit kan heel tijdrovend worden indien veel functies toegelaten zijn. In dergelijke gevallen zijn we beter af met binair zoeken, of zelfs nog beter: hashing (hierbij wordt ieder element opgeslagen aan de hand van een bepaalde sleutel). In mijn voorbeeld heb ik binair zoeken gebruikt. Binair zoeken werkt als volgt: het zoekinterval wordt steeds gehalveerd totdat het correcte element gevonden is. Stel dat we bijvoorbeeld het plaatsnummer (=abscis) van het getal 6 in de volgende gesorteerde rij zoeken:

```
1   2   3   4   5   6   7   8   9
-|---|---|---|---|---|---|---|---|-
2   2   4   6   8   10  12  15  15
```

We beschouwen om te beginnen het interval [1,9]. Het middelste element in dit interval is het (1+9)div2 -de element, dus het 5de element, en dit is 8. Zes is kleiner dan 8, dus moet het in het interval [1,5] liggen. We hebben het zoekinterval dus gehalveerd. Door het interval nu verder te halveren, houden we uiteindelijk maar één element meer over: het gezochte element (indien het aanwezig is tenminste). Op deze manier vinden we als abscis van het getal 6 de waarde 4.



Nog één laatste opmerking: indien een willekeurig aantal spaties tussen de formule-elementen toegelaten is, moeten we er voor zorgen dat we eerst alle spaties verwijderen uit onze uitdrukking. Verder moeten alle letters naar hoofdletters (ofwel kleine letters) omgezet worden. Dit wordt gedaan door de routines REMOVE_BLANKS en CAPITALS.

Het vertalen van het algoritme naar een programma zou nu niet te veel problemen meer mogen opleveren. In listing 3 staat een dergelijk programma afgedrukt. Om de zaken niet al te veel te compliceren, wordt er geen error-controle uitgevoerd, en zijn enkel gehele getallen toegelaten. De routines zullen dus waarschijnlijk wat aangepast moeten worden indien u ze in uw eigen programma's wilt gebruiken. De routines moeten slechts eenmaal ontworpen worden, en kunnen dan steeds ingelast worden als een programma ze nodig heeft. Tenslotte vindt u in listing 4 een voorbeeld van een programma dat de vorige routines gebruikt. Hierin wordt gevraagd om een functie in te geven, en het programma drukt dan de functiewaarden af voor de getallen 1 tot en met 20. (Alle routines zijn ontworpen met TURBO PASCAL).

Appendix: de Backus-Naur-formulering (BNF)

De BNF is een schematische voorstelling die toelaat op een compacte en ondubbelzinnige manier syntactische beschrijvingen te noteren.
De volledige BNF notaties zien er als volgt uit:

<naam> **benoemt een syntactisch element van de taal, dat op zijn beurt een BNF gedaante heeft. vb: <cijfer>**

== **is te lezen als: "wordt gedefinieerd als".**

(...) **duidt een verzameling aan waaruit zal moeten geselecteerd worden.**

| **scheidt de elementen van een verzameling, en is te beschouwen als een selectieoperator.**

[ ] **omsluiten een facultatieve uitdrukking, m.a.w de uitdrukking mag, maar hoeft niet voor te komen.**

{ } **drukken een repetitie operator uit: de omsloten uitdrukking mag nul of meerdere malen herhaald worden.**

tekst (vette druk). Duidt het letterlijk voorkomen aan van het symbool "tekst" in de uitdrukking, en moet overgenomen worden op exact dezelfde wijze als geschreven.

Voorbeeld:

<geheel getal>== <teken> <cijfer> {<cijfer>}

<teken> == (+ | -)

<Cijfer> == (1|2|3|4|5|6|7|8|9|0)

(Bron: Programmeren en het gebruik van de computercursus gedoceerd aan de RUG door prof. dr. ir. G. Hoffman). Langs deze weg wil ik graag Professor Dr. Ir. G. Hoffman (Rijksuniversiteit Gent-België) bedanken voor zijn hulp in de voorbereiding van dit artikel.

Hieronder een overzicht van de bijbehorende listings. De eerste twee listings werden gepubliceerd in Commodore Info nr.1 van deze jaargang.

LISTING 1: PASCAL equivalent van het algoritme EVALUATE.

LISTING 2: TYPE-definities nodig voor de twee procedures.

LISTING 3: zet een uitdrukking om van de infix- naar de postfix-notatie.

LISTING 4: testprogramma (TURBO-PASCAL)

Listing 3: zet een uitdrukking om van de infix- naar de postfix-notatie

```
(*
    REVERSE POLISH  1.4       ---          26 april 1987

    Auteur        : Peter Heinckiens

    Beschrijving : Deze routine zet een mathematische expressie om
                   van de infix-notatie naar de postfix-notatie, ook
                   de reverse polish-notatie genoemd.

    Aanroeping    : reverse_polish (expressie, stack, n)

                    expressie : de om te zetten uitdrukking
                                type : FORMULA_TYPE

                    stack     : hierin komt het postfix-equivalent
                                van de uitdrukking
                                type : POLISH

                    n         : aantal elementen in expressie.
                                type : INTEGER
*)

PROCEDURE reverse_polish (uitdrukking : FORMULA_TYPE;
                          VAR stack   : POLISH;
                          aantal      : INTEGER);

CONST aantal_functies = 2;
      funlen = 5;

TYPE funcinput = STRING [funlen];

VAR eindestack : POLISH;
    pointer    : INTEGER;
    sym        : ARRAY [char] OF SYMBOL;
    fct        : ARRAY [1..aantal_functies] OF FUNCINPUT;
    funcsym    : ARRAY [1..aantal_functies] OF FUNCTIONS;

PROCEDURE remove_blanks;

VAR i,j : INTEGER;

BEGIN

  i := 1;
  WHILE i <= aantal
    DO
      IF uitdrukking [i] = ' '
        THEN
          BEGIN
            FOR j := i TO aantal-1
              DO
                uitdrukking [j] := uitdrukking [j+1];
            aantal := aantal - 1;
          END
        ELSE
          i := i+1

END;

PROCEDURE capitals;

VAR i  : INTEGER;

BEGIN

  FOR i := 1 TO aantal
    DO
      IF uitdrukking [i] in ['a'..'z']
        THEN
          uitdrukking [i] := chr ( ord('A') + ord(uitdrukking[i])- ord('a')

END;

PROCEDURE initsym;

BEGIN

  sym ['+'] := plus;
  sym ['-'] := minus;
  sym ['*'] := times;
  sym ['/'] := slash;

  funcsym [1] := dec;        fct [1] := 'DEC  ';
  funcsym [2] := inc;        fct [2] := 'INC  ';

END;

PROCEDURE expressie;

TYPE charset = SET of CHAR;

VAR  reserve : INTEGER;

  FUNCTION  getsymbool (pointer : INTEGER) : SYMBOL;

  VAR  x : CHAR;

BEGIN

  x := uitdrukking [pointer];
  getsymbool := sym [x];

END;

FUNCTION  nog_elementen (verzameling : CHARSET) : BOOLEAN;

VAR  x : CHAR;

BEGIN
  x := uitdrukking [pointer];
  nog_elementen := (x in verzameling) and (pointer <= aantal);
END;

PROCEDURE  update_next (x : KIND);

VAR hulppointer : NEXTPOINTER;

BEGIN

  new (hulppointer);

  IF eindestack.next <> nil

    THEN   eindestack.next^.link := hulppointer
    ELSE   stack.next             := hulppointer;

  hulppointer^.value    := x;
  hulppointer^.link     := nil;
  eindestack.next       := hulppointer;

END;

          PROCEDURE  stockeer_bewerking (b : SYMBOL);

          VAR hulppointer : OPERPOINTER;

          BEGIN

            new (hulppointer);

            IF eindestack.oper <> nil

              THEN   eindestack.oper^.link := hulppointer
              ELSE   stack.oper             := hulppointer;

            hulppointer^.value    := b;
            hulppointer^.link     := nil;
            eindestack.oper       := hulppointer;

            update_next (operand);

          END;

          PROCEDURE stockeer_getal (x : INTEGER);

          VAR hulppointer : NUMBPOINTER;

          BEGIN

          new (hulppointer);

          IF eindestack.numb <> nil

            THEN   eindestack.numb^.link := hulppointer
            ELSE   stack.numb             := hulppointer;

          hulppointer^.value    := x;
          hulppointer^.link     := nil;
          eindestack.numb       := hulppointer;

          update_next (number);

          END;

          PROCEDURE operator (reserve : INTEGER);

          VAR b : SYMBOL;

          BEGIN

            b := getsymbool (reserve);
            stockeer_bewerking (b)

          END;

          PROCEDURE term;

          VAR reserve : INTEGER;

                  PROCEDURE factor;

                      PROCEDURE getal;

                     VAR x                : INTEGER;
```

```
    FUNCTION omzetting : INTEGER;

    (*
      zet string om naar numeriek getal
    *)

    VAR n, totaal : INTEGER;
        x         : CHAR;

    BEGIN

      totaal := 0;
      x := uitdrukking [pointer];

      n := ord(x) - ord('0');

      WHILE  nog_elementen (['0'..'9']) DO
        BEGIN
          totaal  := 10*totaal + n;
          pointer := pointer + 1;
          x       := uitdrukking [pointer];
          n       := ord(x) - ord('0')
        END;

      omzetting := totaal

    END;

BEGIN    (getal)

  x := omzetting;          (* zet om van char naar integer. *)
  stockeer_getal (x);

END;
   PROCEDURE variabele;

   BEGIN
     IF not (uitdrukking [pointer+1] in ['A'..'Z']) or (pointer=aantal)
       THEN
         update_next (parameter);
         pointer := pointer + 1
   END;

   PROCEDURE wiskundige_functie;

   VAR f           : FUNCTIONS;
       hulppointer : FUNCPOINTER;

      FUNCTION getfunction : FUNCTIONS;

      VAR f      : FUNCINPUT;
          i, l, r,
          teller : INTEGER;

         FUNCTION test (x : CHAR) : BOOLEAN;

         BEGIN
           test := uitdrukking [pointer] = x
         END;

      BEGIN   (getfunction)

         f := '      ';
         teller := 1;

         WHILE not test ('(') DO
           BEGIN
             f [teller] := uitdrukking [pointer];
             teller     := teller + 1;
             pointer    := pointer + 1
           END;

         l := 1;  r := aantal_functies;

         REPEAT
           i := (l+r) div 2;
           IF f <= fct [i]  THEN r := i-1;
           IF f >= fct [i]  THEN l := i+1;
         UNTIL l>r;

         IF l-r = 2 THEN getfunction := funcsym [i]

      END;

   BEGIN   (term)

     factor;

     WHILE nog_elementen (['*','/']) DO
       BEGIN
         reserve := pointer;
         pointer := pointer + 1;
         factor;
         operator (reserve);
       END

   END;

  BEGIN     (expressie)

    term;

    WHILE nog_elementen (['+','-']) DO
      BEGIN
        reserve := pointer;
        pointer := pointer + 1;
        term;
        operator (reserve);                    (*  plus operator  *)
      END

END;


(*  Hoofdprocedure  :  reverse_polish  *)

BEGIN                                      BEGIN
                                             numb := nil;
  pointer := 1;                              oper := nil;
  remove_blancks;                            func := nil;
  capitals;                                  next := nil
  initsym;                                 END;

  WITH eindestack DO                     expressie;

                                     END;

   BEGIN  (wiskundige_functie)

     (* lees functie *)

     f := getfunction;
     pointer := pointer + 1;


     (* lees + verwerk argument *)

     expressie;
     pointer := pointer + 1;


     (* stockeer functie *)

     new (hulppointer);

     IF eindestack.func <> nil

       THEN    eindestack.func^.link := hulppointer
       ELSE    stack.func            := hulppointer;

     hulppointer^.value    := f;
     hulppointer^.link     := nil;
     eindestack.func       := hulppointer;

     update_next (functie);


   END;

BEGIN  (factor)

  CASE uitdrukking [pointer] OF

    '('        :  BEGIN
                    pointer := pointer + 1;
                    expressie;
                    pointer := pointer + 1
                  END;

    'X'        :  variabele;

    '0'..'9'   :  getal;

    '+', '-'   :  stockeer_getal (0);

    'A'..'Z'   :  wiskundige_functie;

  END  (* case *)

END;
```

```
LISTING 4 : testprogramma (TURBO-PASCAL)

PROGRAM test_function (INPUT, OUTPUT);

{$Itype.fct}            (* Las de drie vorige
{$Ipolish}
{$Ievaluate}                           routines in

VAR tekst : FORMULA_TYPE;
    i, y  : INTEGER;
    stack : POLISH;
    a     : CHAR;

BEGIN

  tekst := '';

  write ('f(x) = ');
  readln (tekst);

  i := length (tekst);                (* i het aantal
                                            karakters
  reverse_polish (tekst, stack, i);

  FOR i := 1 TO 20
    DO
      BEGIN
        y := evaluate (stack, i);
        writeln (i, y : 4)
      END

END.
```

AMIGA boekenoverzicht (voorjaar 1988)

Naam:	Schrijver:	Prijs	Taal:
3d Grafikprogrammierung Amiga (+disk)	Jennrich	ƒ 69,--	Duits
68000 Assembler		ƒ 95,--	Engels
68000 Book		ƒ 79,50	Engels
Advanced AmigaBasic	Halfhill, Brannon	ƒ 67,50	Engels
Advanced AmigaBasic	Myers	ƒ 45,--	Engels
Advanced C primer		ƒ 99,--	Engels
Amiga der film	Spanik	ƒ 59,--	Duits
Amiga 1st computer		ƒ 67,50	Engels
Amiga 3d		ƒ 49,90	Nederlands
Amiga 500 für einsteiger	Spanik	ƒ 49,--	Duits
Amiga assembly language programming	Commander	ƒ 39,--	Engels
Amiga basic in & out		ƒ 79,50	Engels
Amiga C for beginners		ƒ 67,50	Engels
Amiga C für einsteiger	Schaun	ƒ 49,--	Duits
Amiga C in beispielen (+disk)	Huckert, Kremser	ƒ 85,--	Duits
Amiga DOS express		ƒ115,--	Engels
Amiga DOS manual		ƒ 95,--	Engels
Amiga DOS handboek	Kerkloh Tornsdorf	ƒ 59,90	Nederlands
Amiga handbook		ƒ 95,--	Engels
Amiga hardware reference manual	Commodore	ƒ 65,--	Engels
Amiga Intern	Dittrich, gelfand	ƒ 79,90	Nederlands
Amiga intuition reference manual	Commodore	ƒ 65,--	Engels
Amiga Jahrbüch	Commodore	ƒ 23,50	Duits
Amiga maschinensprache	Dittrich	ƒ 59,--	Duits
Amiga programmers guide		ƒ 67,50	Engels
Amiga ROM kernal reference manual	Commodore	ƒ 89,--	Engels
Amiga ROM kernal reference manual exec	Commodore	ƒ 65,--	Engels
Amiga tips en truc	Weltner, Hornig	ƒ 69,90	Nederlands
Amiga tips & tricks	Weltner, Hornig	ƒ 67,50	Duits
Amiga User's guide		ƒ 67,50	Engels
Amiga's programmers handbook volume 1	Mortimore	ƒ 65,--	Engels
Amiga's programmers handbook volume 2	Mortimore	ƒ 65,--	Engels
Amiga's user guide to graphic, sound	Myers	ƒ 49,--	Engels
Anwender handbuch		ƒ 45,--	Duits
Balance of power book		ƒ 49,50	Engels
Becomming an Amiga Artist	Sanders	ƒ 55,--	Engels
Beginners guide Amiga		ƒ 67,50	Engels
C primer plus		ƒ 99,--	Engels
C programmers graphics book		ƒ 79,50	Engels
Compute's Amiga applications	Flynn	ƒ 45,--	Engels
Compute's Amiga basic	Halfhill, Brannon	ƒ 45,--	Engels
Compute's Amiga programmers guide	Levy	ƒ 45,--	Engels
Compute's AmigaDos reference guide 1.2	Levitan, Leemon	ƒ 45,--	Engels
Compute's beginners guide to the Amiga	McNeill	ƒ 45,--	Engels
Compute's first book of the Amiga	Compute	ƒ 45,--	Engels
Computer animation		ƒ 95,--	Engels
Das Amiga 500-buch	Breuer	ƒ 59,--	Duits
Das Amiga augsteigerbuch	Stellmach	ƒ 49,--	Duits
Das grosse buch zu AmigaDos	Kerkloh, Tornsdorf	ƒ 59,--	Duits
Das grosse Amiga 2000 buch	Rugheimer, Spanik	ƒ 69,--	Duits
DeLuxe grafik met dem Amiga	Breuer	ƒ 59,--	Duits
Developers reference guide		ƒ 67,50	Engels
DOS handbuch Amiga		ƒ 45,--	Duits
DOS manual book		ƒ 79,50	Engels
Elemantary Amiga basic	Regena	ƒ 39,--	Engels
Flight sim. adv. book		ƒ 67,50	Engels
Flying flight simulator		ƒ 45,--	Engels
Gebruikers handboek Amiga 500-1000-2000	Lawrence	ƒ 39,50	Nederlands
Grafik auf dem Amiga	Kohlen	ƒ 59,--	Duits
Hardware reference manual		ƒ 95,--	Engels
Het Amiga basic boek	Rugheimer, Spanik	ƒ 59,90	Nederlands
Inside Amiga graphics	Keenon	ƒ 67,50	Engels
Inside the Amiga		ƒ 99,--	Engels
Inside the Amiga with C	Berry	ƒ 55,--	Engels
Intuition reference manual		ƒ 95,--	Engels
Kernal libs & devs		ƒ129,50	Engels
Kids and the Amiga	Carlson	ƒ 67,50	Engels
Master am DOS buch		ƒ 67,50	Duits
Music Through midi		ƒ 79,50	Engels
Postscript lan. reference		ƒ 89,50	Engels
Postscript tutorial		ƒ 69,50	Engels
Programmer's guide to the amiga	Peck	ƒ 65,--	Engels

vervolg amiga boekenoverzicht

Programmier h-buch 500-1000-2000 (+disk)Kremser		*f* 85,--	Duits
Programmieren mit AmigaBasic	Henning	*f* 69,--	Duits
Programming graphics on the Amiga	Metcalf	*f* 49,--	Engels
Puppy love		*f*115,--	Engels
Supergrafiek boek voor de Amiga	Jennich, Trapp	*f* 89,90	Nederlands
The Amiga Images, Soun and Animation	Boom	*f* 59,--	Engels
The Amiga book		*f* 79,50	Engels
The Amiga DOS manual 2nd ed. DOS 2.1	Commodore	*f* 59,--	Engels
The Amiga MicroSoft basic progr. guide	Sanders	*f* 55,--	Engels
The Amiga system		*f* 65,--	Engels
The kickstart guide to the Amiga	Ariadne Software	*f* 59,--	Engels
User guide to G & S		*f* 75,--	Engels
Using Aniga DOS		*f* 67,50	Engels
Using the LuxePaint	Anzovin	*f* 49,--	Engels
Wegwijs op de Amiga	Spanik	*f* 59,90	Nederlands

De meeste boeken zijn verder ook te bestellen via de betere computerzaken en boekhandels.

Vragen van gebruikers

Willekeur

Van tijd tot tijd bereikt ons de vraag over de afwijkingen in het werken met RANDOM-getallen in Commodore Basic ten opzichte van het inmiddels min of meer standaard geworden GWBasic-dialect. Zo ook de vraag van de heer van Wuyckhuise, die vraagt *hoe je nu het beste met het commando RND() kunt omspringen.*

Het antwoord op deze toch wel vaak gestelde vraag is vrij eenvoudig te geven. Maar het is altijd goed om ook van de andere kant eens wat meer te lezen over een bepaald onderwerp.
De Basic-opdracht RND() is in feite een functie, waarbij een RANDOM (willekeurig) floating-point getal wordt gegenereerd door de computer, met een waarde tussen 0 en 1. RND is vooral nuttig bij het werken met simulaties, statistische programma's en spelletjes.
De manier waarop RND kan worden gebruikt is eenvoudig. Door tussen de haakjes een getal te plaatsen wordt de functie RND geactiveerd. De **waarde** van dit (floating point) getal is niet van belang, alleen het **teken** (0,+ of -) heeft een functie bij het randomizen. En daarin zit ook de grootste afwijking van Commodore ten aanzien van andere Basic-dialecten.
We onderscheiden drie manieren om RND te gebruiken:

```
10  I = RND(-1)
20  Y = RND(0)
30  X = RND(1)
```

waarbij ook de volgorde het één en ander uitmaakt. De drie verschillende RND-regels geven alle een 'willekeurig' getal retour. Alleen is deze willekeur bij alledrie verschillend.

Het laatste RND-getal wordt door de Commodore op een bepaald adres in het geheugen bewaard. Aan de hand van dit getal kunnen daarna al of niet willekeurig gekozen getallen door RND worden gemaakt. De drie expressie-mogelijkheden maken hier op een aparte manier gebruik van.
De eerste mogelijkheid: RND(-1) zorgt ervoor, dat er een bepaald, vastgesteld getal op die geheugenplek wordt gezet. RND(-2) geeft een ander getal en RND(-100.44) weer een ander. Maar als we over een week opnieuw RND(-1) intypen zullen we hetzelfde resultaat krijgen. De wijze om met een negatief getal de RND-functie te activeren noemen we een 'SEED'. Met dit moedergetal kunnen we een bepaalde serie RND() getallen opwekken, die weliswaar onbepaald zijn, maar waarvan de volgorde van de getallen iedere keer dezelfde zal zijn. Heel goed dus voor simulaties.
Nadat met RND(-GT) de 'SEED' in het computergeheugen is gezet, kunnen we met hulp van RND(+GT) de rest van de serie getallen, geënt op de SEED gaan generen.
Voor RND(0) geldt een andere formule, en daarbij hebben we geen inspraak in een identieke volgorde en weten we ook de waarde van de aangemaakte RND-getallen niet. RND(0) is dan ook de beste oplossing voor spelletjes, omdat in dit geval de 'SEED' wordt bepaald door de tellers in de interne Commodore VIA-klok. En deze tellers, die zo'n miljoen keer per seconde wijzigen hebben daardoor op elk moment een bijna niet te voorspellen waarde. De hiermee gemaakte serie getallen kan daardoor ook op geen elke manier nogmaals in dezelfde vorm worden herhaald.

Om RND-getallen in een bepaalde range te krijgen, bijvoorbeel tussen 0 en 100 kunnen we de volgende formule hanteren:
Range tussen X en Y
A = X + RND(0)*(Y-X)

Machinetaal op de C-16

De heer Eggink uit Wychen schrijft in z'n brief, dat het onjuist is om de cursus machinetaal alleen te schrijven voor de gebruikers van een C-64. Hij vraagt *of het niet mogelijk is om ook voor de C-16 een aparte cursus te publiceren.*

De machinetaal, zoals we die kennen voor de 65XX serie processoren, is een bijzonder lage taal. Dat wil zeggen, dat we in deze taal slechts beschikken over een kleine hoeveelheid instructies, die bovendien voor iedere computer die met een microprocessor uit de 65-serie is uitgerust gelijk zijn.
Alle kennis uit de cursus van Tjipke van der Land kan zondermeer worden toegepast op de C-16, evengoed als op de C-64 en zelfs op een Apple//.
Het gaat in deze cursus om het geven van algemene informatie over de manier waarop er in machinetaal kan worden geprogrammeerd. Om dit zo practisch mogelijk te kunnen doen is besloten de C-64 als uitgangspunt te nemen. Maar als er hier en daar wat computer-adressen worden aangepast werken de voorbeelden evengoed (en snel) op de C-128, Plus/4 en de C-16, terwijl zelfs de oude PET en CBM nog mee kunnen doen.

Out of Memory

Uit Breda komt een bericht van J. van Noord, met betrekking tot het runnen

van overgetypte programma's op z'n C-16. Naast de conversie van diverse commando's, blijkt hij tamelijk vaak een 'OUT OF MEMORY' error op z'n scherm te krijgen. Dit gebeurt ook met de listing van de database uit de cursus Basis Basic.

De reden waarom een OUT OF MEMORY op het scherm verschijnt is erg simpel. De computer beschikt niet over voldoende RAM-geheugen om het hele Basic-programma, compleet met alle adres(geheugen)ruimte voor de verschillende variabelen te bevatten. Een echte 'syntax-fout' is het dus niet. Daarom is het ook niet nodig de hele listing op onvolkomenheden te controleren, want daarligt het niet aan. De ruimte die een dergelijk programma in beslag neemt is gewoon te groot. Deze fout treedt natuurlijk sneller op bij de kleinere computers en de C-16 is inderdaad niet erg royaal waar het geheugenruimte betreft. Vooral het werken met arrays vergt nogal wat overhead-geheugen, waardoor de beschikbare ruimte snel (te) vol wordt.
Als oplossing voor dit probleem kunt u natuurlijk het beste een RAM-uitbreiding voor de C-16 aanschaffen; want (voor zover nog beschikbaar) zijn die inmiddels erg goedkoop geworden. Lukt dit niet dan zal er iets aan de oorspronkelijke listing gedaan moeten worden. Dit kunt u doen, door de listing waarbij de fout voorkomt, vooral in het begin nauwkeurig te onderzoeken. Daar vindt u meestal de DIM-opdrachten, die voor de arrays RAM-ruimte reserveren. Nu is die ruimte ook nodig om het programma te laten runnen, maar er hoeft meestal niet zoveel te worden gereserveerd als de programmeur in gedachten had. In het geval van de Basis Basic listing vindt u dergelijke opdrachten bijvoorbeeld in regel 130 en 160/170:

DIM A$(300,5), Z$(10) en
DIM ST(LOG(300)/LOG(2)+4),1)

Als u enige vaardigheid hebt in het interpreteren van Basic, ofwel de cursus bij dit programma hebt gelezen, zult u weten, dat er in dit geval RAM-ruimte wordt gereserveerd voor de pointers van maximaal 300 adressen. Dit aantal is gebaseerd op een vrije RAM-ruimte van ongeveer 32.000 Bytes. De C-16 heeft niet zoveel aan boord, dus zult u in regel 130 en 170 het getal 300 moeten wijzigen in 100. Daarmee schept u een maximum capaciteit van 100 adressen, maar het programma werkt naar behoren.
Om verder RAM-(geheugen)ruimte vrij te maken, kunt u ook het programma enigszins strippen. Dit kan, door bijvoorbeeld alle REM-opdrachten eruit te laten. Iedere letter in REM-regels neemt per slot van rekening Byte-geheugen in beslag en er staan ter verduidelijking vaak veel REM-regels in Basic-programma's. Vergeet daarbij niet ook een originele listing met REM-opdrachten te bewaren, zodat later bij eventuele wijzigingen op deze listing kan worden teruggevallen.

Star printers

Het fenomeen printers blijft de gemoederen bezighouden. Niet vreemd, want wat kun je nu nog doen zonder een goede printer. De heer Van De Heuvel uit Utrecht schrijft een prachtige brief op een STAR NL 10, waarin hij iets uit de doeken doet over z'n problemen met de aansturing.

We hebben al veel geschreven over het besturen van met name de niet-Commodore printers. Er zijn echter zoveel verschillende types en modellen, die samen met een overweldigende hoeveelheid aangepaste Kernals en cartridges ervoor zorgen dat de C-64 toch een standaard output krijgt. Niet erg elegant, maar in veel gevallen werkbaar. Ook mijn advies bij de aanschaf van een (nieuwe) printer voor de Commodore computers luidt zondermeer: 'Koop een standaard Centronics (parallel) printer, zodat deze periferiek ook later bij eventueel grotere systemen te gebruiken is. Als interface tussen beide zijn er veel mogelijkheden, waarbij je wel moet denken aan de grafische mogelijkheden. We kunnen en willen in deze rubriek geen expliciete merken noemen, maar probeer bij aanschaf de complete configuratie in de winkel uit, samen met eigen software, zodat later geen verassingen tevoorschijn komen.
De heer Van De Heuvel liet aan het einde van z'n brief in 5 regels zien, hoe een STAR NL 10 kan worden gebruikt om met de C-64 toch erg kleine letters op papier te toveren zijn, die we zonder commentaar overnemen:

```
10  OPEN 4,4
20  PRINT#4,CHR$(27)
    CHR$(83)CHR$(48);
30  PRINT#4,CHR$(27)CHR$(15);
40  PRINT#4,CHR$(27)CHR$(51)
    CHR$(16);
50  PRINT#4,"Commodore INFO
    etc...."
60  CLOSE 4
```

Adressen C64/C128

Er is veel vraag naar het converteren van programma's van de C-64 naar de C-128. Hoewel de 128 probleemloos functioneert in 64 mode wil men toch de nodige complexe Basic-software en machinetaal overzetten.

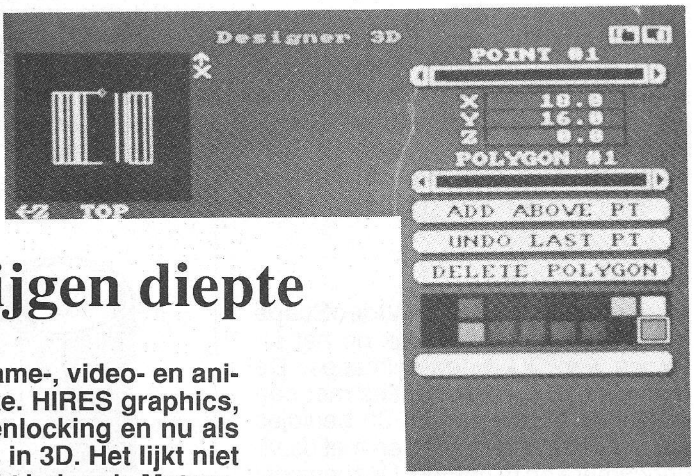
In het bestek van deze rubriek kunnen we helaas niet al te diep ingaan op de materie, daarvoor ontbreekt de ruimte. Wel zullen we een paar veel gebruikte KERNAL/BASIC interpreter adressen van de C-64 naast de C-128 leggen, zodat de verwoede programmeur aan de slag kan. Voor zover ons bekend zijn de hierna genoemde entree-adressen van beide machines volledig compatibel:

Naam	C-128	C-64
TXTTAB	2D-2E	2B-2C
LINNUM	16-17	14-15
VARTAB	2F-30	2D-2E
ARYTAB	31-32	2F-30
MAXMM1	39-3A	37-38
VARNAM	47-48	45-46
VARPNT	49-4A	47-48
STATUS	90	90
FNLEN	B7	B7
LA	B8	B8
SA	B9	B9
FA	BA	BA
BCOLD	4000	A000
INIT	4023	E394
CONT	4B34	A857
STOP	4BCB	A82C
READY	4D2A	C474
INLIN	4F93	A560
FNDLIN	5064	A613
LINGET	50A0	A96B
LIST	50E2	A69C
NEW	51D6	A642
RUNC	51F3	A659
CLEAR	51F8	A65E
RETURN	5262	A8D2
DATA	528F	A8F8
PRINTN	553A	AA80
CMD	5540	AA86
PRINT	555A	AAA0
DIM	587B	B081
RUN	5A9B	A871
ISVAR	7978	AF28
FRE	8000	B37D
VAL	804A	B7AD
PEEK	80C5	B80D
RND	8437	E097

We zijn natuurlijk lang niet volledig in deze lijst, maar als begin kun je toch behoorlijk wat aanpassingen verrichten, terwijl gelukkig de sequentie in de Basic-ROM niet erg gewijzigd is tussen beide types, waardoor je gemakkelijk ook de tussenliggende routines zult kunnen opzoeken. De vectoradressen op PAGE 3 zijn voor beide computers gelukkig identiek.

Jan Bodzinga

Amiga is een ster in 2D-graphics en -animaties. Onlangs verschenen ook de eerste drie dimensionale pakketten voor de Motorola 68000 CPU op Nederlandse markt. Wat tot voor kort op microcomputers voor onmogelijk werd gehouden is nu waarheid geworden: Met VideoScape 3D, Sculpt 3D en Forms in Flight wordt de Amiga-monitor een waar 3D-tekenpalet en -animatiefabriek.



AMIGA 3D

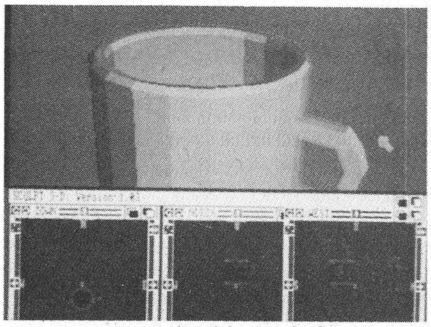
Graphics en animaties krijgen diepte

Wat de Commodore Amiga's voor de grafische, reklame-, video- en animatiewereld betekenen grenst aan het ongelofelijke. HIRES graphics, een palet van 4096 kleuren, flitsen, animaties, video-genlocking en nu als klap op de vuurpijl ook nog eens graphics en animaties in 3D. Het lijkt niet op te kunnen. Voor een "prikkie" aan software en een licht bejaarde Motorola 68000 CPU bereikt men hetzelfde als hetgeen tot voor kort slechts met vele tonnen kostende apparatuur mogelijk was.

Na Deluxe Paint en -Video en Aegis Animator komen nu in een snel tempo 3D-pakketten voor de Amiga's op de markt. Wij bekeken Forms in Flight, VideoScape 3D en Sculpt 3D.
De eerste 3D-graphics en -animaties van de Amiga verschenen in 1986. Allen Hastings, een Amerikaanse hobbyist, slaagde er in om Amiga 3D-frames te maken en deze op 16 mm speelfilm vast te leggen. De verkregen resultaten waren zo fantastisch dat het bekende Amiga software-huis Aegis Development direkt met Hastings in zee ging. Het eindproduct werd het krachtige animatiepakket **VideoScape 3D.** Concurrent Electronic Arts kon zo iets natuurlijk niet op zich laten zitten en bracht een nieuwe Deluxe Paint-versie uit met 3D-optie.
Inmiddels hebben de softwarehuizen Micro magic met **Forms in Flight** en Byte by Byte met **Sculpt 3D** eveneens het Amiga 3D-pad betreden. Vermoedelijk zullen op korte termijn ook andere softwareleveranciers volgen.

Forms in Flight

Onder de titel "vormen in vlucht" brengt het Amerikaanse Micro Magic een uitgekiend 3D-tekenpakket op de markt. **Forms in Flight** lijkt wel wat op een 3D CAD-programma. Naast gewoon 3D-tekenen kan de gebruiker ook de ruimtelijke coördinaten invoeren. Verder staan de CAD-technieken hidden line removal, wire-frames en solid shading ter beschikking.
Om Forms in Flight goed te kunnen benutten zijn 1 MB en liefst meerdere MBs aan vrij RAM onmisbaar en een harde schijf gewenst. In 1 MB gaan 600 vormen met elk vier zijden. Elke MB meer doet daar nog eens ruim 1.700 oppervlaktes bij.
Elk object wordt eerst uit één basisvorm, de polygoon, opgebouwd. Er zijn 3 typen: spline, freehand en regular. Naar keuze kan er direkt in 3D of eerst in 2D getekend worden. In de praktijk blijkt dikwijls dat eerst in 2D tekenen en daarna tot 3D uitbreiden sneller en nauwkeuriger dan direkt uit de vrije hand driedimensionaal schetsen.
Is eenmaal een basisobject ontworpen dan kan dit object door middel van het verwijderen van de hidden lines,



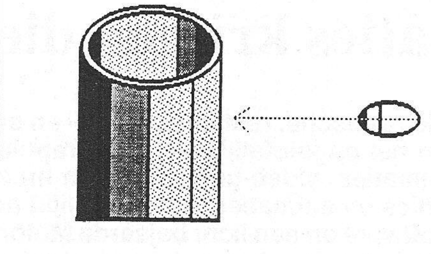
Een voorbeeld van de 3-D mogelijkheden van de Amiga

solid shading (uitvullen van de vlakken in schaduwtinten) bewerkt of met gelijksoortige objecten tot een grotere structuur gecombineerd worden. Verder kent Forms in Flight de tekentools geheel of gedeeltelijk kopiëren, beeldspiegelen en rotaties. De rotaties kunnen simpele pan- of rolbewegingen maar ook ruimtelijk zijn. De toeschouwer kan het object dan uit alle ruimtelijke gezichtspunten bekijken. Desgewenst kan er ook in/uitgezoomd worden.
Met standaardfiguren uit vorm-bibliotheken kan de kunstenaar op deze wijze gecompliceerde figuren opbouwen. Alleen de eigen creativiteit en ruimtelijk inzicht stellen hierbij hun 3D grenzen.
Forms in Flight werkt met de schermresoluties 640 x 200 pixels of 640 x 400 pixels in 2, 8 of 16 kleuren. De afmetingen van de 3D-figuren blijven bij een eventuele verandering van schermmodus behouden.
De **animaties** worden gedefiniëerd door het aantal frames en de bewegingsrichting(en) in deze frames per tijdseenheid of beeldjesreeks. Eerst kiest de gebruiker een object uit en activeert dan één van de bewegingsmenu's. Met behulp van deze trekmenu's definiëert de artiest het aantal frames en bijvoorbeeld het aantal graden en de as bij een rotatie. Het is mogelijk om meerdere animatieblokken aan elkaar te koppelen of meerdere objecten onafhankelijk van elkaar in één en dezelfde sequentie te animeren.
Forms in Flight is volledig trek(intuition)menu- en muisgestuurd. Dat helpt een heel stuk, maar verwacht niet de eerste keer meteen al verbazingwekkende 3D-vormen te kunnen construeren. Geduld is hier een schone zaak. Zeker bij het animeren, saven en afspelen van gecompliceerde beelden. Afdrukken kan ook op een HP-GL of compatibele plotter met een

maximum oplossend vermogen van 10.240 x 6.400 pixels. Helaas zijn deze plotters te duur voor de doorsnee hobbyist en blijken matrix screendumps niet mogelijk. Een ander gemis is het ontbreken van IFF-files voor opname in andere grafische software voor de Amiga.
Voor real time video-opnamen beschikt Forms in Flight over het afspeelprogramma **Fast Flight** dat het Amiga RAM en de animatie- algoritmen optimaal benut.
Forms in Flight gaat in de winkel vermoedelijk zo'n *f* 180,- kosten.

VideoScape 3D

Alleen hastings (Aegis) **VideoScape 3D** richt zich voornamelijk op het afspelen van 3D video-animaties. De bedoeling is korte seqenties met een maximale framerate van 30 beeldjes per seconde af te spelen en met de videocamera op te nemen. Desgewenst is ook frame voor frame opname mogelijk om de beeldjes op professionele video-apparatuur of 16 mm speelfilm over te zetten.
Voor dat u denkt dat het allemaal een makkie is even een waarschuwing: het maken van geanimeerde logo's, videotitels en animaties van enkele seconden kost hoogstens een regenachtige zondagmiddag. Voor het maken van een tekenfilmpje van vijf tot 10 minuten mag men daarentegen toch wel een weekje uittrekken. Het gaat allemaal aanzienlijk sneller dan met de ouderwetste sheets, maar het kost altijd nog meer tijd dan je denkt om een goede animatiefilm te maken!
VideoScape 3D plakt object-files aan bewegingsfiles. Daarna moet de gebruiker nog aangeven uit welke hoek de camera filmt en hoe deze camera zelf beweegt (draait). Met deze drie filestappen in het achterhoofd zijn verbazingwekkend goede animaties te verwezenlijken.
Het uitgangspunt van de object-files zijn weer standaardvormen zoals cylinders, doosjes en kegels. Een **Easy Geometry Generator** (EGG) helpt de artiest daarbij. Helaas gaat deze EGG stapsgewijs en onomkeerbaar te werk. De software vraagt de artiest om een aantal beschrijvingen van het object en voert de antwoorden zonder mogelijkheden tot herstel uit. Gaat het mis dan moet alles weer overnieuw. Bovendien laat het programma het gemaakte object pas zien als deze vorm af is. Is het object eenmaal klaar dan kan het gesaved of voor het ontwerpen van een nieuwe vorm gebruikt worden.
Het **Object Composition Tool** (OCT) laadt meerdere objecten in het RAM om hen onderling te combineren en te redigeren. De edit-mogelijkheden zijn o.a. verandering van grootte, positie, oriëntatie en kleuren. Alle gecombineerde en/of veranderde objecten kunnen vervolgens naar één objectfile worden weggeschreven.
Het 3D-deel van VideoScape zit in de **Designer 3D**. Deze 3D-ontwerper is in feite een apart subprogramma binnen de VideoScape-mantel. Oorspronkelijk heet dit programma **ROT** en was een ontwerp van Colin French.



Gebruik van Shading

Met de editor kan de artiest 98 x 98 punts polygonen ontwerpen door de overeenkomstige puntcoördinaten in te voeren. Het object kan door drie verschillende vensters bekeken worden. Een misser vinden wij dat ROT slechts ROT-vormen wil inladen (VideoScape-formats worden geweigerd) en behalve het polygoon geen standaardvormen kent.
Het combineren van objecten met de OCT blijkt soms een moeizaam karwei. Snijdende objecten kan VideoScape niet automatisch aan en men zal dergelijke complexe figuren met de hand moeten editen. Daarvoor zijn wiskundige kennis en geometrisch inzicht dringend gewenst. Bij minder gecompliceerde figuren nemen de software routines daarentegen wel veel werk uit handen. Naar keuze kan men de complexe figuren als RAM-sparende draadmodellen (wire-frames) of met schaduwwerking (solid shading) laten zien.
De video-animaties worden gedefiniëerd door de aard en snelheid van de beweging samen met het bijbehorende camerastandpunt vast te leggen. Er zijn twee typen beweging: positioneel (van punt A naar B in de 3D-ruimte) en draaien om een (X-,Y-, Z)-as zonder daadwerkelijk van positie te veranderen. Sleutelframes leggen het begin en eind van de bewegingssequentie vast. De tweens (aantal tussenliggende frames) bepalen de vloeiendheid en/of snelheid van de beweging. Zowel de object- als de camerabeweging worden met een ASCII-text editor in een motion- file opgeslagen.
De commando-centrale van VideoScape 3D is het main screen of Commando-venster. In dit window staan drie aparte vensters: het object description-, het camera motion- en het viewing options- venster. Het object description window dient voor het laden van geometrische en bewegingsobjectfiles. Tot de trukages behoren metamorfose (het ene object gaat door vormverandering in het andere over) en twee dezelfde objecten in één scene. Ook kan de gebruiker hier op de hand de bewegingscoördinaten invoeren.
Camera motion hanteert de camerabeweging en het in/uitzoomen. Tot slot het viewing options-venster voor de 4 verschillende resoluties van 352 x 220 tot 704 x 440 pixels en full overscan. Verder is het mogelijk om met behulp van dit venster IFF voor- en achtergronden te laden.
Een player-programma zorgt voor de playback als single frame (voor idem dito VCRs) en als Anim file voor een continue videoregistratie.
VideoScape 3D is een animatie-programma voor de professional met veel geduld en wiskundig inzicht. Deze gebruikergroep kan met videoScape verbazingwekkende effecten uit de Amiga slepen. Een ander punt is dat het pakket eisen aan de hardware stelt: minimaal 1 MB aan boord en een harde schijf. Hopelijk worden de door ons gesignaleerde editor-problemen en handmatige invoer van complexe structuren in een volgende versie aangepast. De prijs ligt momenteel rond de *f* 400,-.

Sculpt 3D

Eric Grahams **Sculpt 3D** vindt zijn oorsprong in een Amiga-demo van een jonglerende robot. Deze robot stond in een soort schaakbordlandschap met drie spiegelende ballen te jongleren die door een "3D-verlichting" tot echt driedimensionaal leven kwamen. Men noemt deze 3D-verlichtingstechniek ook wel **ray tracing**.
Het nu door Byte by Byte uitgebrachte Sculpt 3D is een 3D- tekenpakket en bevat geen animatie-software. Wel is het mogelijk om de gemaakte objecten later in andere animatieprogramma's op te nemen. Byte by Byte heeft er zelf een aantal van deze routines als public domain software vrijgegeven.
Eén van de sterkste kanten van Sculpt 3D is de relatief eenvoudige werking van de 3D-object-editor. Het hoofdscherm bevat drie vensters, Byte by Byte spreekt van een tri-view. De drie

windows laten het object respectievelijk in een noord-zuid-, oost-west- en een top- bodem-oriëntatie zien. Alle zichtbare modellen worden als wireframes weergegeven.
De vensters bieden de normale Amiga sizing box-, front/back- en drag bar-opties. Extra, dus van Sculpt 3D-origine, zijn:

° Vier pijlen voor het scrollen van de objecten

° Zoom in/out

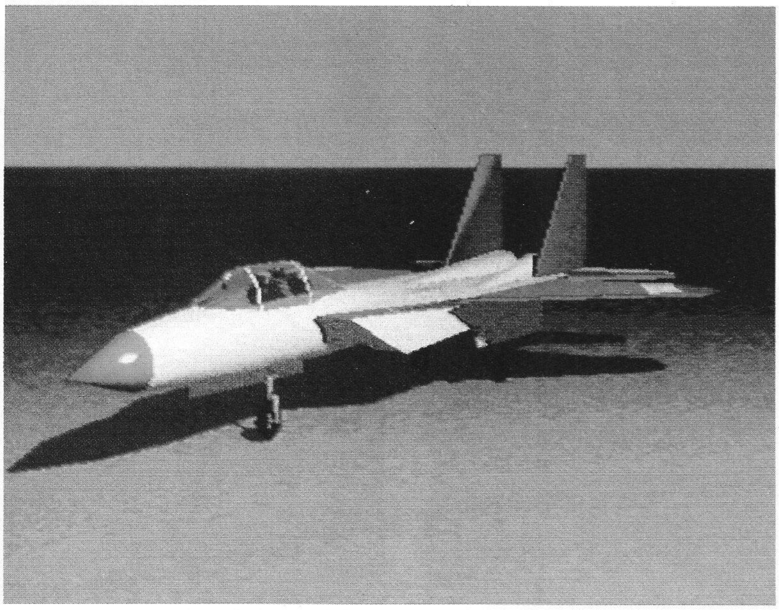
° En centreeroptie voor de cursor in het venster

Het opbouwen van objecten gaat met de muis. De gebruiker geeft een drietal punten aan en een driehoek-tekenutility maakt daar vervolgens een driehoek van. Dit driehoekig oppervlak wordt weer gebruikt voor het samenstellen van complexe geometrische figuren. Behalve het uit de vrije hand tekenen is het eveneens mogelijk om via een venster coördinaten in te voeren.

Cuve-tool

Een cuve-tool wordt gebruikt om een aantal met elkaar verbonden punten te creëren. Daar het invoeren van coördinaten en losse punten nogal wat tijd vergt biedt Sculpt 3D de gebruiker staandaard al een aantal geometrische figuren zoals kegels, cylinders, kubussen, prisma's en hemisferen. Samen met dupliceren en spiegelen zijn vele constructies mogelijk. Het kern object blijft echter het driehoekige oppervlak waar ook de kant en klare figuren zijn opgebouwd.
Zoals bij een goed tekenpakket hoort voorziet Sculpt 3D in het verfijnen van de ontworpen objecten. Voor ronde oppervlaktes zijn er de Sphere- en Smoothing-tools die het van origine hoekige vlak vrijwel rond maken. Iets wat bijvoorbeeld VideoScape niet lukt. Voor het tekenen zelf gaat Sculpt 3D uit van een observator die uit een bepaalde richting en onder een bepaalde hoek naar het te creëren object kijkt. De grootte van het gezichtsveld wordt bepaald door de beeldhoek van de objectieven op de fictieve camera van de observator. Ook voorziet dit 3D tekenprogramma in het automatisch opstellen van meerdere lichtbronnen (voor ray- tracing) en het tekenen van achtergronden. Er zijn verschillende tekenmodussen:

- De **frame-modus** is de simpelste. Eenvoudige modellen kosten niet meer dan enkele seconden tekentijd. Bij gecompliceerdere figuren kan men al gauw op een uurtje of wat ontwerptijd rekenen.



Sculpt 3D

- In de **painting**-modus gebruikt Sculpt 3D gekleurde veelhoeken en shading door middel van het richten van gekleurde lichtbronnen. In deze modus is slechts één kleur per oppervlak mogelijk.

- De **snapshot ray-tracing**-modus berekent de kleur van elke schermpixel op grond van de reflectie van (gekleurde) lichtstralen. In deze modus zijn nog geen shading-technieken mogelijk.

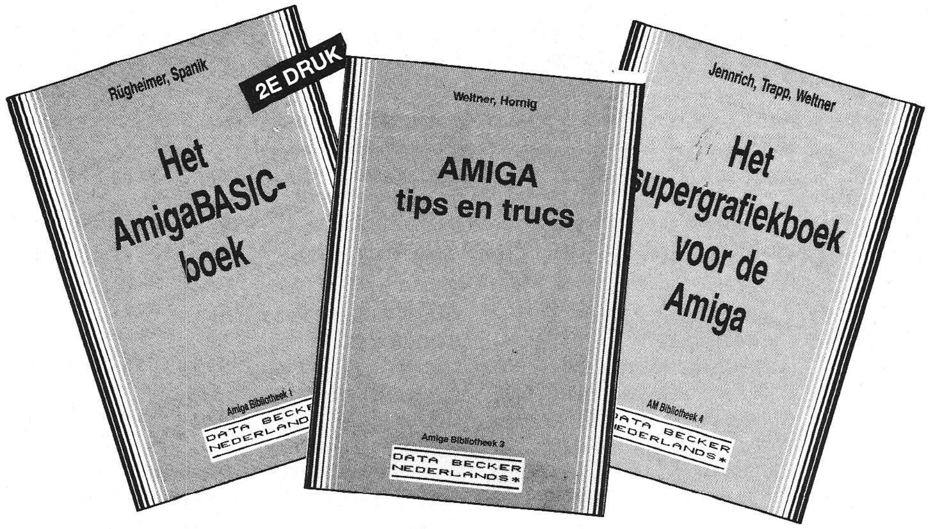
- De **photo ray-tracing**-modus is gelijk aan de snapshot-modus, maar kan wel levensechte schaduwen ontwerpen.

De beide ray-tracing-methodes gebruiken helaas erg veel rekentijd. Bij gecompliceerde figuren staat de Amiga zo ettelijke uren te number crunchen. Byte by Byte helpt de gebruiker met een speciale batch-modus waarbij de artiest kan aangeven welke objecten en/of achtergronden achter elkaar opgetekend moeten worden en het aantal scenes. Sculpt 3D gaat dan verder automatisch aan de slag en slaat elk gereed beeld op de (harde) schijf op.
Wat de schermmodus betreft kan de gebruiker kiezen uit LORES, HIRES, interlaced en non-interlaced. Maximaal zijn er in de Hold And Modify (HAM)-modus 4096 kleuren in 6 bit planes beschikbaar.
Andere interessante mogelijkheden van Sculpt 3D zijn o.a.: Het **grabber-tool** voor het verplaatsen van een willekeurig aantal opgegeven punten binnen één object. De grabber grijpt alle punten die binnen zijn bereik liggen. De **Magnet** trekt sterker aan de punten die dichtbij de muiscursor gelegen zijn dan de punten die op grote re afstanden liggen. Net als een echte magneet kan dit tool ook punten afstoten. Het **unslice**-tool maakt van over elkaar liggen oppervlaktes één geheel.
Sculpt 3B behoort tot de krachtigste tekenpakketten die voor de Amiga verkrijgbaar zijn. Mits de gebruiker over het nodige geduld en artistieke talenten beschikt en de Amiga uitrust met 1 MB en harde schijf zijn imposante creaties mogelijk. Versie 1.1 (meer dan de helft sneller bij het raytracen dan de oude 1.0- uitvoering) kost circa *ƒ* 200,-. Als enige nadeel geldt het gebrek aan interne animatietools.

De Amiga gaat met 3D een nieuw grafisch tijdperk in. Rare roodgroen-brilletjes en dure electronica zijn niet meer nodig om een real life driedimensionaal effect op de monitor te tekenen. Bovendien bieden twee van de hiervoor besproken pakketten nog eens de mogelijkheid om de zelf gemaakte 3D-frames te animeren.
Natuurlijk is het maken van 3D-platen en -animaties niet voor iedereen weggelegd. Werkelijk professionele resultaten kosten veel tijd en vereisen een goed inzicht in de lastige materie van coördinatenstelsels, bekijkhoeken (views), belichtings- en shading-technieken. Binnen een uurtje komt de leek niet verder dan een simpel wireframe-model. De ware artiest kan daarentegen met een simpele Amiga met voldoende geheugen en een harde schijf aan boord tot realistische 3D-beelden en/of animaties komen die op een minicomputer of mainframe niet zouden misstaan.

U.S

Nieuwe AMIGA boeken



De AMIGA is een computer met ongekende mogelijkheden, dat dit ook een keerzijde heeft is de gebruikers al wel bekend. Hoe meer mogelijkheden een computer heeft, hoe moeilijker het is om al deze mogelijkheden te gebruiken. Dit is een van de voornaamste redenen dat de Amigaboeken van Data Becker in het Nederlands vertaald zijn.

Wegwijs op de AMIGA

Het boek begint daar waar alle gebruikers beginnen, met het uitpakken van de computer. In tegenstelling met de handleiding wordt hier goed uitgelegd waar alle aansluitingen voor dienen. Er wordt uitgebreid ingegaan op begrippen als intuition, multitasking pulldownmenu's en de preferences. Jammer is het dat er voor zo'n boek de schermfoto's letterlijk zijn overgenomen uit de Duitse boeken. Het is toch wel storend hier duitse teksten te zien staan. Alle workbench funkties worden behandeld waarbij veel gebruik wordt gemaakt van voorbeelden. Omdat toch een flink aantal mensen de bij de AMIGA geleverde handleiding niet goed kunnen begrijpen, zou eigenlijk dit boek standaard bij elke nieuwe AMIGA moeten worden bijgeleverd.

Titel:	**Wegwijs op de Amiga**
Auteur:	**Spanik**
Uitgever:	**Bruna & Zn Utrecht**
ISBN	**9022934136**
Prijs	***f* 59,90**

Het AmigaBasicBoek

AmigaBasic is er voor iedereen, beginner, gevorderde en zelf een professional heeft er mee te maken. Stap voor stap worden alle geheimen van de basic onthuld. Computerannimatie, IFF files, sequentiële en relatieve bestanden. Het boek is onderverdeeld in een aantal hoofdzaken. Het grafisch werken met de amiga, de amiga en de gegevensbestanden en geluiden op de amiga. Een belangrijk hoofdstuk is gewijd aan de foutmeldingen, en wat nog belangrijker is: hoe zijn deze te voorkomen. Erg makkelijk in het gebruik is het basic-naslaggedeelte. Hier in zijn alle AmigaBasic instructies ingedeeld in functiegroepen en daarbinnen alfabetisch gerangschikt. Buiten de instructie is ook de syntax en een uitleg opgenomen. Een aantal bruikbare listings completeren het geheel.

Titel:	**Het AmigaBasic boek**
Auteur:	**RÜgheimer, Spanik**
Uitgever:	**Bruna & Zn Utrecht**
ISBN	**9022934187**
Prijs	***f* 59,90**

Het SuperGrafiekBoek voor de Amiga

De Amiga is bij uitstek een computer voor grafische toepassingen, getuige de vele teken- en presentatieprogramma's. De mogelijkheid om tot 4096 verschillende kleuren op het scherm te zetten, sprites en bobs te gebruiken, spreken tot een ieders verbazing. Het boek is zowel voor de basic- als de C-programmeurs te gebruiken. Er staan een groot aantal voorbeelden in. Er wordt een uitgebreide beschrijving gegeven van het grafische systeem van de Amiga (view, viewport, rasterport, de opbouw van de bitmaps, screens en windows).

Titel:	**Het SuperGrafiek boek**
Auteur:	**Jennnrich, Trapp, Weltner**
Uitgever:	**Bruna & Zn Utrecht**
ISBN	**9022934705**

Amiga tips en trucs

Het boek is in twee gedeeltes gesplitst. In het eerste deel wordt er gewerkt met AmigaBasic van MicroSoft. De nadruk ligt op de bestaande System Software Module. In dit deel zijn onder andere de routines genomen voor de verschillende lettertypes, screendump routines, grafische afbeeldingen op het scherm. In het tweede gedeelte komen de gevorderde programmeurs aan hun trekken en komt de taal C aan de orde. Met behulp van voorbeelden worden de principes van de opbouw van de Amiga theoretisch uitgelegd en ondersteund met een aantal programma's. Zelfs worden er gebruiksmogelijkheden behandeld die in geen enkele handleiding voorkomen.

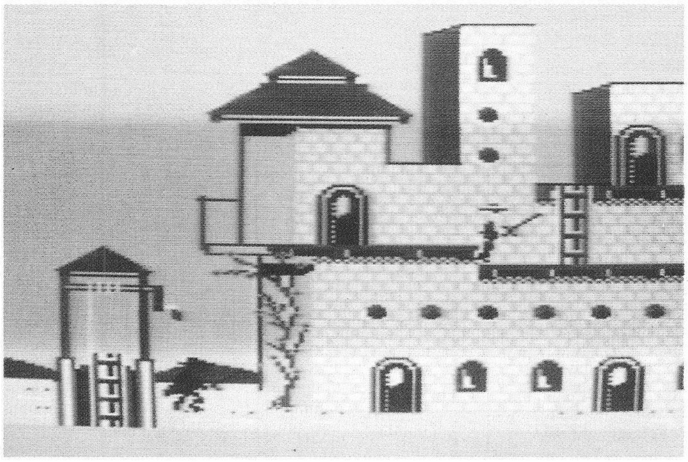
Titel:	**Amiga tips en trucs**
Auteur:	**Weltner, Hornig**
Uitgever:	**Bruna & Zn Utrecht**
ISBN	**902293442x**
Prijs	***f* 69,90**

Rob Goudriaan bespreekt weer enkele klassiekers onder de computerspelletjes voor de Commodore 64.

Oud van Goudriaan

Zorro

Zorro is door de firma Datasoft uit zijn winterslaap gehaald. De hoofdfiguur is een overbekend personage uit het verre Amerika. In vroegere tijden nam hij het altijd op voor de arme en zwakkere mensen. Deze gemaskerde man is een zeer populaire televisieheld, en ook in strips is hij erg geliefd. Het kenmerk van Zorro is zijn eigen teken een -Z-. Omdat hij een uitstekend zwaardvechter was liet hij dit teken altijd achter op zijn slachtoffers. Nu is het dan jouw beurt om in de huid van deze held te kruipen en schone dames te redden. Het spel begint als je als Zorro uit de lucht valt in een plaatsje midden in een grote dorre woestijn. Overal hangen er soldaten rond, kijk hier goed voor uit. In een flits is er nog te zien dat een dame wordt ontvoerd. Het enige dat er dan nog rest is de gevallen zakdoek op te pakken en op het hulpgeroep te reageren. De opdracht is: redt deze dame uit de handen van de nietsontziende bandieten. Het zakdoekje kan alleen maar opgepakt worden door vanaf het balkon op de waterput te springen, en daarna in een vliegende vaart de dame en de bandieten achterna. Maar voordat de achtervolging kan worden ingezet moeten er een aantal opdrachten worden uitgevoerd. Daal in de put af, je komt dan in een ruimte waar drie ballen liggen en deze liggen er niet voor niets. Om naar de overkant te komen zijn deze ballen de enige mogelijkheid gebruik ze dus goed. Eenmaal aan de overkant aangekomen ga je naar rechts tot je een glas ziet, dat je moet meenenemen. In totaal moet je drie voorwerpen verzamelen. Om bij het glas te komen moet je eerst een voorwerp op het platform neerzetten. Dat gaat dan omlaag en het scherm waarachter het glas staat gaat omhoog. Je moet nog verder afdalen, een plant ophalen, deze is misschien zwaar genoeg om het balletje aan het rollen te brengen (dat klinkt geheimzinnig). Maar al met al kan je nog niet bij een glas, dit moet van boven af gebeuren. dus maar weer terug in de put. Na dit geheel volbracht te hebben ga je op zoek naar een schatkamer. Deze kamer zul je nog vele malen moeten betreden, want hier zijn zeer veel voorwerpen opgeslagen die je op je missie nodig hebt. Een van de eerste dingen die je nodig hebt is een sleutel. Anders is de deur een hindernis die niet te nemen is. In de kerk ga je naar de bovenste verdieping. Springen, zodat je een fles kunt pakken. De fles moet bij de plaatselijke kroeg worden ingeleverd, geef hem aan de man aan de bar. Doordat deze met enkele slokken het laatste bodempje uit de fles drinkt, en al het nodige ophad, valt hij als een blok in slaap. Zijn (dikke) buik is als springschans te gebruiken. Door het hangen aan een lamp gaat er een geheim luik open. Hierin moet je afdalen. Hier blijkt dat je eindelijk het glas kunt pakken. We moeten weer omhoog om naar de andere voorwerpen te gaan zoeken.



En zo komt men bij een smederij, in de kerktoren, en kom je uiteindelijk uit bij een open graf, waarin je moet afdalen, nadat je drie de voorwerpen hebt verzameld.

En dan zijn we aan het moeilijkste deel van deze missie gekomen, alle goudzakken die je ziet moeten worden verzameld. Hierbij moet je goed opletten dat je niet verdwaalt, want de weg terug vinden is lastig en kost veel tijd. In dit ondergronds gedeelte blijken zeer vele kamers te zitten, touwladders, bewegende vloerdelen vergen veel van je klimkunsten. Niet verdwaald en alle goudzakken verzameld, dan zie je boven in beeld ineens de drie voorwerpen verschijnen. Via deze voorwerpen kun je naar boven klimmen. Dit lukt niet voordat je de drie ballen in de linker kamer in het water hebt gegooid. Eenmaal boven aangekomen kom je door een gevangenis, hierin zitten een aantal arme, onschuldige mensen. Deze moeten worden gered. Ook hier echter geen spoor van de ontvoerde dame (of was je dit in het vuur van het spel vergeten). Als alle gevangenen zijn bevrijd ben je aangekomen op de binnenplaats van een kasteel waar de dame is gevangen. Heb je alle bewakers ontlopen en denk je dan deze schone dame in je armen te kunnen sluiten dan heb je het mis. Want net als je je ogen even hebt gesloten wordt je van de muur afgegooid, vertwijfeld vraag je je af wat je nu weer hebt fout gedaan. Ten koste van een leven moet je weer terug naar de schatkamer. Via een roos, de put, de gevangenis en de binnenplaats kun je de dame bevrijden, als tenminste niet alle moed je in de schoenen gezakt is.

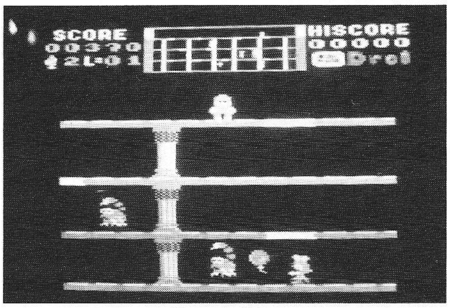
Hard en Mack

In dit spel staat u als bouwvakker centraal. Er staat u een zeer zware klus te wachten. De bouwvakvakantie is voorbij en er blijkt van alles mis te zijn. Het bouw- en woningtoezicht staat op uw vingers te kijken. Mijdt deze ambtenaar want bij een aanraking heeft hij een dodelijke uitwerking. Aangezien u maar drie levens heeft is dus een uiterste voorzichtigheid geboden. Het spel bestaat uit drie verschillende levels en ieder level heeft zijn eigen problemen en moeilijkheden. Eén van de eerste moeilijkheden is een kamer met gaten in de vloer. De opdracht luidt hier deze gaten te dichten. De balken waar dit mee moet gebeuren liggen gelukkig niet al te ver weg, ze

bevinden zich op dezelfde verdieping. Als deze opdracht tot een goed einde is gebracht moeten ze nog even vast worden geschroefd. Hiervoor moet u de ronddollende schroef gebruiken. Maar kijk uit want dit lijkt gemakkelijker dan het is, tenslotte loopt ook de lastige ambtenaar nog steeds rond. Tot overmaat van ramp gaat een spijkermachine stuk. Hij spuugt alleen maar spijkers in het rond. De aanraking met deze spijkers is ook al dodelijk. Gelukkig staan er wel wat hulpmiddelen tot je beschikking. Een lift aan de linkerkant van de stelling, een trampoline rechts zijn hier enige voorbeelden van. Via de touwen kan er naar boven of beneden worden geklommen. Zijn alle balken op zijn plaats en aangedraaid dan ga je naar het tweede level. Op veld twee wordt het zo mogelijk nog zwaarder, hier staan Mack de bouwvakker weer extra moeilijkheden te wachten. Minder handige collega's hebben hier allerlei losse stukken gereedschappen laten liggen. Het lastige heerschap van bouw- en woningtoezicht staat er echter op dat alles opgeruimd is. Natuurlijk ligt het een en ander niet op de makkelijkste plekken. De ambtenaar is hier wel wat makkelijker te omzeilen want zijn route door deze etage is steeds dezelfde. Dus even kijken hoe hij loopt en dan kun je je eigen route daar aan aanpassen. Een op en neer gaande balk is het hulpmiddel om een etage hoger te komen. In veld drie staan er weer een groot aantal verrassingen voor je klaar. De blokken die hier liggen moeten in een machine worden gestopt. De bekende trampoline kan ook hier weer worden gebruikt om de ambtenaar te ontwijken. Dit is en blijft een lastig mannetje. Als deze drie levels zijn doorlopen ben je eigenlijk alweer aan een nieuwe vakantie toe. Voor de doorzetters, de echte computerfanaten, begint het spel hier weer van voren af aan. Alleen zijn er nu twee zeer lastige ambtenaren bijgekomen die al je doen en laten observeren.

Bruce Lee

Alweer een filmheld met een hoofdrol in een computer spel. Dit maal kruipen we in de huid van Bruce Lee. In de film is hij onverslaanbaar, dit in tegenstelling tot dit spel. Hier is hij juist erg kwetsbaar. Voortdurend wordt hij dwars gezeten door twee vijanden. Het zijn ook niet de minste tegenstanders: een groene Yamo en een zwarte Ninja. Deze figuren laten je, want je bent in de huid van Bruce gekropen, niet met rust. Dit maakt je taak er niet gemakkelijker op. Wat is nu eigenlijk je opdracht? In negentien verschillende kamers hangen lampions en olielampen, deze moeten door jou verzameld worden. Deze lampen zijn alleen te pakken door er tegen aan te springen. Hierbij moet je ook nog op je tegenstanders letten, je kan ze gevoelige klappen en trappen uitdelen. Hun aanvallen zijn af te slaan, maar gemakkelijk zul je het niet van ze winnen. Soms moeten er in een kamer eerst alle lampen worden verzameld voor er ergens een geheim luik opengaat. Als je het spel start begin je in een tuin. Na de lampen verzameld te hebben gaat er in de tweede tuin een luik open, je daalt hierdoor naar een lager gelegen gedeelte en hier begin-



Bruce Lee

nen de problemen pas goed. Buiten de twee lastposten kost het een perfecte timing en een groot doorzettings vermogen om hier verder te gaan. Sommige lampen kan je niet bereiken als je er de eerste keer komt. Deze kun je dan alleen via een omweg, of als je er later terugkomt pakken. Ben je uiteindelijk de laatste hindernis voorbij en denk je: ik ben er, dan heb je het mis want er moeten nog erg veel lampen en olielampen worden verzameld. Zelf met de hindernissen is het nog niet afgelopen. Bij een volgende gelegenheid moet je laten zien hoe goed je bent in het klimmen in de touwen. Er staan je ook nog ontploffende struiken, dodelijke spiezen en nog veel meer van dit soort werk te wachten. Een van de gevaarlijkste dingen zijn ongetwijfeld de lopende banden en de spijkerbedden. Deze zijn echt niet om uit te rusten van alle inspanningen. Zoals eerder gezegd zijn niet alle lampen direkt te pakken het is dus een goede zaak te noteren waar en in welke kamer men lampen heeft laten hangen, zodat je later nog (even) terug kan gaan. Een plattegrond bewijst hierbij uitstekende diensten. Het is dan ook bijna onmogelijk de eerste keer dat men dit spel speelt het tot een goed einde te brengen. Elke keer dat men speelt zal men een stukje dichter bij de oplossing kunnen komen. In sommige velden moet men een bepaalde volgorde aanhouden anders is het onmogelijk er later nog bij te kunnen.De laatste kamer is de troon kamer van de grote tovenaar en hier hangt nog een lantaarn. Lukt dit je dan brandt het kasteel tot de grond af. en begint het spel weer van voren af aan.

Drol

Eindelijk dan een schietspel in deze gouwe ouwe rubriek. Het spel heet Drol (ik kan het ook niet helpen). Bij dit spel gaat het niet zoals bij zo veel spelen om op ruimteschepen te schieten, of planeten aan te vallen. Hier gaat het echt om de zelfverdediging. Je moet je verdedigen tegen rondzwervend ongedierte. De opdracht bij dit spel is je broertje en zusje te redden. Zij zijn op stap gegaan met hun hagedis en krokodil en waarschijnlijk zijn ze hierbij verdwaald. Ma is als eerste op zoek naar de kinderen gegaan, maar ook zij is verdwaald. Dus ben jij de aangewezen persoon om op zoek te gaan. Je bent zeer modern aangekleed, een soort ruimtepak met een straalmotor op je rug. Verder heb je alleen de beschikking over een pistool. Het spel bestaat uit drie delen, je start op de bovenste etage van een gangenstelsel. Je kunt lopen en vliegen. Door de luiken kun je naar een andere verdieping. Allereerst moet je je zusje redden, in deze ronde moet ook de hagedis worden gered. Allerlei enge griezels proberen je dit te beletten. In de tweede ronde moet je je broer(tje) gaan redden. Als extra moet er ook voor de krokodil worden gezorgd. In deze ronde zijn je tegenstanders sterke magneten. Dit maakt het geheel er niet makkelijker op. Springende monsters en een tovenaar complementeren het geheel. Zijn je broertje en zusje gered dan rest nog je moeder, ook zij wil graag naar huis. Hierbij zijn de moeilijkheden vooral te verwachten van speren, vliegende bijlen en zelfs grote slangen. Ik zal niet alle verrassingen verklappen want er moet nogwat overblijven. De grootste verrassing komt echter als je denkt het spel tot een goed einde gebracht te hebben. Als je hier meer over wilt weten zul je op zoek moeten in de diverse computerzaken om dit spel te bemachtigen.

Wederom hebben we veel inzendingen ontvangen uit het hele land. Zonder inzendingen kan deze rubriek niet bestaan. Daarom is de volgende mededeling van groot belang: in verband met een verhuizing is ons adres gewijzigd. Stuur in het vervolg je briefjes en je Micro's naar: Postbus 43048, 1009 ZA in Amsterdam.

Basic Micro's



Computers rekenen anders dan mensen. Ze werken met het binaire talstelsel. Het omzetten van decimaal naar binair is niet moeilijk, maar ook niet echt gemakkelijk. Het omrekenen is vaak wel nodig als we bijvoorbeeld de video- of de geluidschip willen aansturen, of als we een sprite of een karakter ontwerpen.

Omrekenen

Het eerste programma voor de C-64 lost het omrekenen voor ons op.
Het is ingezonden door Coen Antens uit Rucphen, en het is ook goed te gebruiken met zijn programma uit het vorige nummer. Na het uitvoeren van het programma kan het omrekenen gebeuren met: SYS 49152,getal. Het getal moet dan tussen 0 en 255 liggen.

```
10 FORI=0TO44:READQ:POKE49152+I,Q:C=C+Q:
NEXT:IFC=5919THENPRINT"DATA OK"
20 DATA32,253,174,32,158,183,134,251,169
,128,141,16,192,162,0,169,0,36,251,208
30 DATA5,169,48,76,28,192,169,49,32,210,
255,110,16,192,232,224,8,208,232,169
40 DATA13,32,210,255,96
```

Smooth-Scroll

Als er een prijs uitgeloofd wordt voor regelmatig insturen, dan wint Obbe Vermey die. We ontvingen van hem dit keer onder andere een programma voor de C-64 dat een supergaaf doorlopend tekstbalkje onderin het beeldscherm tovert. We doen hierbij nogmaals het verzoek aan de inzenders om een controle-tellertje in te bouwen bij Micro's met machinetaal, want anders kan intikken en RUNnen weleens een op hol slaande computer opleveren.

```
10 FORT=0TO121:READA:POKE49152+T,A:C=C+A
:NEXT:IFC<>15399THENPRINT"DATAFOUT":END
20 FORT=0TO255:POKE49408+T,T:NEXT:POKE49
261,255:SYS49152
30 DATA120,169,31,141,20,3,169,192,141,2
1,3,169,129,141,26,208,173,17,208,41
40 DATA127,141,17,208,169,242,141,18,208
,88,96,173,25,208,141,25,208,48,7,173
50 DATA13,220,88,76,49,234,173,22,208,41
,247,13,120,192,141,22,208,162,144,202
60 DATA208,253,173,22,208,41,248,9,8,141
,22,208,206,120,192,173,120,192,16,37
70 DATA169,7,141,120,192,160,,185,193,7,
153,192,7,200,192,39,208,245,172,121
80 DATA192,185,,193,141,231,7,200,192,25
5,208,2,160,,140,121,192,76,188,254,1,1
```

Nog enkele opmerkingen:

- Het getal dat in 49261 gePOKEd wordt in regel 20 geeft aan hoe lang de tekst is (maximaal 255).
- De tekst wordt in het geheugen gezet vanaf adres 49408, door middel van de schermcodes.
- De tekstbalk op het scherm is niet beschermd tegen tussentijds wissen door PRINTCHR$(147) of doordat er iets anders overheen gezet wordt.
- Het duurt ongeveer vijf seconden voor je de tekst ziet.

Rechthoeken

Peter Bovert uit Knokke-Heist (België) stuurde enkele Micro's in voor de C-128. Hij geeft een demonstratie van de BOX-opdracht.

```
10 GRAPHIC1,1
20 BOX1,55,110,65,160:FORL=0TO180STEP5:B
OX1,55+L,110,65+L,160,L:NEXT
30 BOX1,235,110,245,160:FORL=130TO5STEP-
5:BOX0,55+L,110,65+L,160,L:NEXT:GOTO20
```

Amiga-pointer

Tony Smits uit Berkel-Enschot stuurde een Micro voor de Amiga in, die de pointer die door de muis bestuurd wordt, verandert van een pijltje in een handje.

```
FOR x=0 TO 63:POKE 20212+x,0:NEXT X
FOR x=0 TO 35:READ y:POKE 20212+x,y
NEXT x:PALETTE 17,1,.73,0
PALETTE 18,0,0,0:PALETTE 19,1,1,1
DATA 0,0,127,255,127,246,128,15,0,54,127
DATA 207,7,246,8,15,0,54,7,207,1,246,2
DATA 15,0,54,1,207,0,118,0,143,0,0,0,127
```

Oogspel

Het oogspel is een simpel reactiespelletje, waarbij je moet proberen het getal te lezen dat heel kort in beeld verschijnt. Het is ons toegezonden door Philip Rappé uit Knokke-Heist (België).

```
10 PRINTCHR$(147)"RADEN MAAR":A=INT(RND(
1)*200)+1:PRINTA:FORB=1TO20:NEXT
20 PRINTCHR$(147):INPUT"WAT WAS HET";C:P
RINT"HELEMAAL ";:IFC<>ATHEN40
30 PRINT"GOED!":GOTO50
40 PRINT"FOUT!":PRINT"HET WAS";A
50 FORV=1TO1000:NEXT:GOTO10
```

Pie Chart

Een Pie Chart is een taartvormige grafiek. We ontvingen een programma in Simon's Basic van Wilfred van Dijk uit Zeist, dat dergelijke grafieken in 3D tekent.

```
10 COLOUR12,12:PRINTCHR$(147):INPUT"HOEV
EEL GETALLEN";G:DIMW(G),D(G)
20 FORI=1TOG:PRINT"GETAL"I":";:INPUTW(I)
:T=T+W(I):NEXT:T=360/T
30 FORI=1TOG:D(I)=W(I)*T:NEXT:HIRES0,12:
CIRCLE160,100,80,30,1
40 ARC160,125,90,270,12,80,30,1:LINE239,
100,239,125,1:LINE81,100,81,125,1
50 ANGL160,100,0,80,30,1:H=0:FORI=1TOG:H
=H+D(I):ANGL160,100,H,80,30,1
60 IFH>90ANDH<270THENFORJ=100TO125:ARC16
0,J,H,H+1,1,80,30,1:NEXT
70 NEXT:WAIT653,1
```

Kleurensoepje

Ook dit keer leveren wij een bijdrage aan deze rubriek, en wel met een Micro voor de C-64, waarmee eenkleurige schermen voorgoed tot het verleden behoren.

```
10 POKE53281,0:POKE53280,0:FORT=0TO28:RE
ADA:C=C+A:POKE50432+T,A:NEXT
20 IFC=3080THENSYS50432:LIST:DATA120,169
,13,141,20,3,169,197,141,21,3,88,96,238
30 PRINT"FOUT IN DATA":DATA134,2,173,134
,2,77,33,208,41,15,240,243,76,49,234
```

Bloemen

De C-128 is steeds vaker het onderwerp van de inzendingen. Het nu volgende programma is gemaakt door Stanley Appel uit Teteringen. Het tekent een aantal bloemen op het scherm.

```
10 COLOR0,1:COLOR1,2:COLOR4,1:COLOR5,2:F
ORD=1TO5.5STEP.5:X1=160:Y1=100:R=1
20 GRAPHIC1,1:FORA=1TOD*175STEPD:X=R*COS
(A)+160:Y=R*SIN(A)+100:DRAW1,X1,Y1TOX,Y
30 X1=X:Y1=Y:R=R+.5:NEXT:SLEEP4:NEXT
```

Staafdiagram

Maarten Wagemans uit Eindhoven stuurde een Micro voor de C-16 in, die het waarschijnlijk ook doet op de C-128. Van 40 ingevoerde waardes wordt een overzichtelijk staafdiagram getekend.

```
10 A=0:DIMA(40):PRINTCHR$(147):FORL=1TO4
0:PRINT"GETAL";L;" IS";
20 INPUTA(L):IFA(L)<0ORA(L)>199THENPRINT
"FOUT":GOTO20:ELSENEXT:GRAPHIC1,1
30 FORL=1TO40:FORK=ATOA+5:DRAW1,K,199TOK
,199-A(L):NEXT:A=A+8:NEXT
40 FORL=1TO1000:NEXT:GETKEYA$:GRAPHIC0:C
LR:PRINTCHR$(147)
```

Gatenbalk

Iedereen kent wel de tekeningen die voorwerpen laten zien die je niet kunt maken, zoals kubussen, balken en ruimtelijke driehoeken, waarbij ergens een verbinding zit die niet kan bestaan. M. Benjamins uit Alkmaar schreef een Micro die zo'n onbestaand figuur tekent voor de C-128, maar het moet mogelijk zijn om hem om te schrijven voor bijna elke computer.

```
10 GRAPHIC1,1:DRAW,247,53TO240,40TO80,40
TO50,92TO57,105TO217,105TO247,53
20 DRAW,247,53TO88,53TO76,79TO202,79TO20
9,66:DRAW,202,79TO209,92
30 DRAW,87,79TO94,66TO223,66TO209,92TO50
,92:DRAW,88,53TO94,66
40 FORT=0TO2:CIRCLE,125+T*26,60,4,3:PAIN
T,125+T*26,60:NEXT
50 CHAR,4,17,"ZITTEN DE GATEN VAN BINNEN
":CHAR,4,19,"OF VAN VOREN?"
```

Flitsende balken

De "snelheid" van Basic wordt gedemonstreerd met de volgende Micro van de hand van Henk-Jan Boer uit Kockengen.

```
10 PRINTCHR$(147)
20 POKE53280,1:POKE53280,0:POKE53281,1:P
OKE53281,0
30 :::::::::::::::::::::::::::::::::::::
::::GOTO20
```

Van dezelfde auteur komt ook nog een programma in Simon's Basic. Het tekent veelhoeken.

```
10 PRINTCHR$(147):INPUT"HOEVEEL HOEKEN";
H:DH=INT(360/H+.5):HIRES1,0
20 ARC160,100,0,360,DH,120,100,1:FORI=0T
O360STEPDH:ANGL160,100,I,120,100,1:NEXT
30 PAUSE5:NRM
```

Tot slot nog een leuke tip om voor bijna elke Commodore computer (niet voor de Amiga), die listings leesbaarder maakt. Zet eens helemaal vooraan je programma's een regel, waarbij je het volgende letterlijk intikt:

5 REM""[DEL][RVS on][SHIFT-M]E[SHIFT-S][RVS off]LIST

Als je dit intikt, let er dan op dat je de teksten tussen de vierkante haken en de haken zelf niet overneemt. In plaats daarvan moet je de toetsen indrukken die genoemd worden tussen de vierkante haken. Als je alles goed doet, gebeurt er voortaan het volgende als je LIST intikt: de tekstkleur wordt wit, het scherm wordt schoongemaakt, en er wordt LIST bovenaan het scherm gezet, waarna de rest van de listing volgt.

Peter Broekhuizen